Markov Chain Monte Carlo methods for simulation in pedigrees

Cheal, Ryan

Award date:
1996

Awarding institution:
University of Bath

Markov Chain Monte Carlo Methods for Simulation in Pedigrees

submitted by

Ryan Cheal

for the degree of Ph.D.

of the

University of Bath

1997

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

To

My Amazement

Abstract

Dyfal donc a dyrr y garreg.
(Constant blows break the stone)

Motivation

Pedigree analysis is the science of extracting information from family trees or pedigrees. This thesis has concentrated on probability calculations on the pedigree of the Przewalski Horse (PH). We were asked to provide evidence of the effect of different management programs on the pedigree and this has led to developing new efficient simulation methods able to deal with large, complex pedigrees.

Tools

In the past, an exact method called peeling has been used to calculate probabilities on pedigrees, but this method is unsatisfactory for large pedigrees due to its vast memory requirements and its inability to provide uncertainty measures. In the thesis we investigate simulation methods for the calculations.

Stochastic simulation models and a Bayesian approach provide the general framework to model pedigree data and to incorporate prior information obtained for about half of the horses in the PH pedigree. The models are implemented using Markov Chain Monte Carlo (MCMC) algorithms and are structured to be strictly local in nature. This simplifies the algorithms, making them easier to understand and implement. However, they are still computationally intensive.

Application

Firstly, peeling is used in an EM algorithm to obtain the maximum likelihood estimates of the allele frequencies of the founders for traits of two and three alleles of the PH pedigree. However, this method breaks down with more than three alleles for the reasons mentioned above. The Gibbs Sampler, a relatively new method of simulation, forms the basis of the simulating. We replace the peeling with the Gibbs Sampler in the EM algorithm. However,

problems with the irreducibility of the Markov Chain and doubts as to the algorithm's ability to converge in higher dimensional space exist, and so we try to improve on this by using a Bayesian framework as a computational engine. By setting up a constant *Dirichlet* prior we can use the Gibbs Sampler to generate samples from the likelihood which will be proportional to the posterior. Using this framework, estimates for the marginal means and associated statistics are easy to obtain; the remaining problem is how to get good estimates of the MLEs from the samples and various forms of density estimation are investigated. Throughout the thesis we use four decimal places for calculations. For most practical applications this could be considered overkill but we do this in order to assess the accuracy of our algorithms. If less accuracy is required then the run time of the algorithms can be correspondingly reduced.

Visualisation

We will be dealing with traits of up to six alleles and this gives us the problem of how to visualise points and likelihood surfaces of high dimensionality. There are exceptionally powerful visualisation tools and there are others, some well known, that rarely outperform the best ones. In the higher dimensions we use visualisation to check the performance of the Gibbs Sampler and find that a multi-panel display plotting allele pairs gives the best results.

Layout

The thesis is broken down into the following parts:

- Chapter 1 provides an introduction to the subject of pedigree analysis and explains the statistical theory that underlies the techniques used in subsequent chapters.
- Chapter 2 gives the history of the PH pedigree and uses the peeling EM algorithm to provide evidence about the relevance of different management programs.
- Chapter 3 replaces the peeling step of the EM with simulated answers generated by the Gibbs Sampler. These are checked against the exact results for two and three alleles and extended to the higher allele traits.
- Chapter 4 uses a fully Bayesian framework as a computational tool to do the simulation and investigates the difficulties of finding MLEs from the realisations using density estimation.
- Chapter 5 summarises the work and outlines possible areas of further research.

A bibliography and index are included for reference.

Keywords

- Ancestral Inference
- Bayesian Inference
- Density Estimation
- EM Algorithm
- Gibbs Sampler
- High Dimensional Visualisation
- Markov Chain Monte Carlo
- Markov Random Fields
- Pedigree Analysis
- Peeling
- Przewalski Horse
- Relaxation Algorithm
- Simulated Annealing
- Stochastic Simulation

Contents

Chapter 1

Introduction

Could I read it?
If I could read it, did I believe it?
If I believed it, did I care about it?
And if I cared about it
What was the quality of my caring?
And did it last?
(Phillip Larkin 1922–1985)

Genetics is the way that human beings are characterised and pass on these characteristics to their children. We have known for some time that diseases such as haemophilia, cystic fibrosis and colour blindness are solely genetic and are passed from generation to generation, but we are now finding that many more diseases such as cancer, heart problems and diabetes have some kind of genetic component. Consequently, there is a lot of interest in the subject. Two models, biological and mathematical, have been formulated.

1.1 Biological Model

Chromosomes are the biochemical structures that carry genetic information. Each cell in

Figure 1-1: The Double α Helix: The diagram shows the twisting of the strands of DNA into the well-known double α helix structure. Two base-pairs are also shown.

the human body has 23 pairs of chromosomes, of these 22 pairs are called *autosomes* and the remaining pair specifies the sex of the individual. One of each pair is inherited from the mother and one from the father.

They are made up of an informative part, a length of DNA, wrapped around structural proteins. The DNA contains a long sequence of four types of molecules: adenine (A),

thymine (T), guanine (G) and cytosine (C). Each molecule identifies and pairs up with one of the others; A identifies with T and G with C. However as shown in Figure 1-1, these strands are able to twist giving the famous *double α helix*. DNA can duplicate itself. The strands pull apart and the unbounded pairs re-bind with their partner molecules from the pool of loose surrounding molecules.

The set of all 23 pairs of chromosomes is called the *genome* and each pair of molecules A–T or C–G is called a *base pair* or *base*.

The human genome consists of about three billion base pairs. Most bases, approximately 99%, are *homomorphic* – the same in all humans – and roughly 90% are identical for all mammals.

1.2 Mathematical Model

A mathematical model of genetics has been formulated in parallel with the biological model. When Mendel decided to favour his mathematical rather than theological instinct and produced his infamous experiment with pea-pods, he started the whole idea of genetics. His insight into genetic structure over 120 years ago has shown itself to be still valid through all of the advances in the subject.

1.2.1 Terminology

Mendel's First Law of Heredity (Mendel 1965) stated that genetic information is stored in discrete 'packets'.

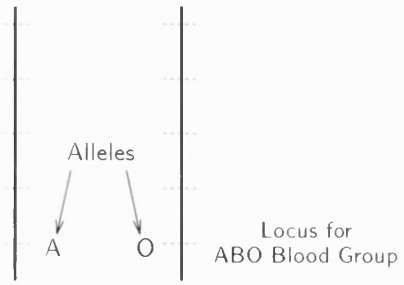


Figure 1-2: PART OF THE HUMAN GENOME: The figure shows the locus for the ABO blood group. The individual has a heterozygous AO genotype which gives rise to a phenotype or blood group a.

Humans are the same as pea-pods in being *diploid* which means that each genetic characteristic, such as hair colour in humans or shell smoothness in pea-pods, is described by two such units. Mendel's 'packets' of information are called *genes* which are linearly arranged along the *chromosomes*. A gene is a sequence of base pairs of varying length. On each chromosome there are many *loci*. A *locus* is the position on the chromosome of a *polymorphism* – that is a position where a base or a sequence of base pairs may show variation. The possible states at a locus are called *alleles* and the two alleles, one on each of a pair of chromosomes, make up the *genotype* at a locus. An individual who carries two copies of the same gene is known as *homozygous*, whilst an individual carrying different

alleles is known as *heterozygous*. The genotype may affect the physiology of the host in the form of some characteristic or *trait*.

The genotype is not directly observable but the trait can be, for example a genetic disease or a property such as the blood groups. The set of all physiological possibilities are called the *phenotypes*. An example showing part of the human genome with with locus for the *ABO* blood group is shown in Figure 1-2. Mendel's laws of inheritance specify the transition of alleles from parents to offspring. If a parent has genotype *AO* then Pr(offspring inherits allele A) = Pr(offspring inherits allele O) = $\frac{1}{2}$. An example pedigree showing Mendelian transmission of genes is shown in Figure 1-3(a).

1.2.2 Pedigrees

In order to make inferences about a genetic trait we need data on a set of related individuals which is called a *pedigree* or *genealogy*. These are collected for various reasons such as historical interest, paternity cases and animal breeding, which is the case that will occupy us in this thesis.

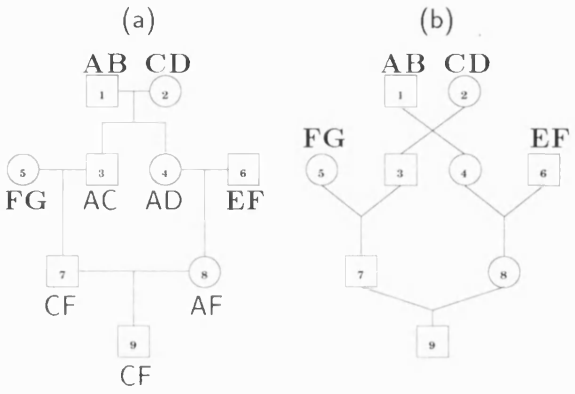


Figure 1-3: EXAMPLE PEDIGREE: Figure (a) shows the traditional format of an example pedigree showing Mendelian transmission of genes. Figure (b) shows a marriage node format of an example pedigree illustrating the genes of the founders only.

A genealogy $\mathcal{G}$ contains the individuals $\mathcal{I}$ and also the relationships between them.

Definition 1.1. An individual is called a *founder* if its parents are not in the pedigree.

The traditional format of representing a pedigree is shown in Figure 1-3(a) and the marriage node format is shown in Figure 1-3(b). Figure 1-3 (a) also shows an example of Mendelian transmission of genes. In large pedigrees where there are many generations, usually some of the phenotypes of individuals at the bottom of the pedigree are known through DNA or blood testing but individuals higher up the pedigree, who were alive before modern testing procedures, yield no phenotypic information.

Although the definition of a pedigree covers a random list from a large population, this is uninteresting to us as relationships would be so distant, and hence untraceable, that all the members of the pedigree would be founders. We are far more interested in a small population descended from a small number of founders where almost everyone is related or a small genetic isolate that has little or no gene exchange with other populations. This is usually either an isolated human population or a captive animal population.

Examples of different pedigrees – a simple, a zero looped and a complex pedigree – are given in Figure 1-4. A simple pedigree is one in which at least one of the parent pair of each offspring in the pedigree is a founder, a zero-loop pedigree is one where both parents of each offspring are in the body of the pedigree but no loops are formed by the marriage node graph and a complex pedigree is one where the breeding creates loops in the pedigree.

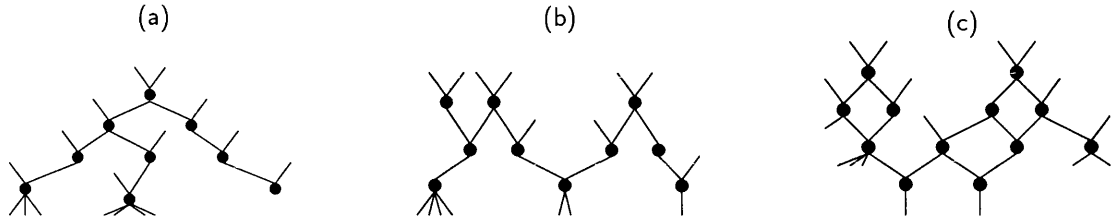


Figure 1-4: EXAMPLES OF PEDIGREES: Plot (a) shows a simple pedigree where at least one parent of each offspring is a founder. Plot (b) shows a zero-loop pedigree where both parents of each offspring are in the body of the pedigree but no loops are formed by the marriage node graph. Plot (c) shows a complex pedigree where the breeding creates loops in the pedigree.

There are several approaches to studying a pedigree. Nowadays we tend to examine genetic traits or genetically linked diseases within the pedigree. For human pedigrees, a small Newfoundland population has been studied by Larsen, Barnard, Buhler, and Marshall (1976) and Thompson (1981), who looked at cancer and immunodeficiency; the Mennonite–Amish community have been studied for proprionic acidemia by Kidd, Wollf, Hsia, and Kidd (1980) and Thompson (1983); and an Eskimo population of Greenland has been investigated by Edwards (1980) and Sheehan (1991).

1.2.3 Genetic Models

The relationship between the underlying, unobservable genotypes and the phenotypes is described by the genetic model. This relationship may be simple or a little more complicated. For example, the blood groups are a three allele trait with alleles A B and O, and genotypes AA:AB:AO:BB:BO:OO. Under a *dominant* model any individual who carries the dominant allele displays its characteristic, while a *co-dominant* model allows all the genotypes to be distinguished phenotypically.

The ABO system is modelled by A and B co-dominant to O, giving phenotypes or blood groups **a**:**ab**:**b**:**o** as AA AO and BB BO are phenotypically indistinguishable. We can alternatively define this model as O *recessive* to A and B.

The following definitions from Thompson (1986) help in defining the genetic model.

Definition 1.2. Let the *transmission probability*, denoted $\tau(i, j, k)$, be the probability that the parents with genotypes j and k produce an offspring with genotype i.

Example 1.1. TRANSMISSION PROBABILITIES: In the ABO blood-group system

$$\tau(AB, AA, BO) = \frac{1}{2} \qquad \tau(OO, AO, BO) = \frac{1}{4}. \tag{1.1}$$

□

Definition 1.3. Let the *penetrance probability*, denoted $\rho(\phi, i)$, be the probability that an individual with genotype i has the observed state, or phenotype, ϕ.

Example 1.2. PENETRANCE PROBABILITIES: Again, using the ABO blood-group system

$$\rho(\mathbf{a}, AA) = 1 \qquad \rho(\mathbf{a}, AO) = 1 \tag{1.2}$$

which indicates that the O allele is recessive with respect to the A allele. □

These penetrance probabilities can be collected and represented in a penetrance matrix.

Throughout this thesis, we shall be examining 16 traits ranging from 2 to 6 alleles and all of them are co-dominant.

1.2.4 Allele Frequencies

Continuing our ABO blood group example, if we let π_1, π_2 and π_3 be the allele frequencies for A B and O respectively and assuming random mating, equal fitness and no selection, then the genotype frequencies will remain constant from generation to generation and the proportion of genotypes AA:AB:AO:BB:BO:OO is in the ratio $\pi_1^2 : 2\pi_1\pi_2 : 2\pi_1\pi_3 : \pi_2^2 : 2\pi_2\pi_3 : \pi_3^2$. This is known as the *Hardy-Weinberg* equilibrium. Random mating assumes you are equally likely to have children by your sister as any other woman, which may be true for animals but is not true for humans. Equal fitness assumes that the average number of offspring doesn't depend on parental genotypes, which will be untrue for lethal traits where the affected host does not survive to reproduce. No selection means that, for example, individuals with an A gene do not hunt out other A individuals to marry.

Definition 1.4. Let the *population frequency*, denoted $\pi(i)$, be the frequency of the i^{th} genotype in the population from which the founders are drawn.

1.3 Likelihood Function for Pedigrees

When computing probabilities on pedigrees the basic statistical tool is the likelihood function

$$L\{\phi(D), G, M\} = P\{\phi(D)|G, M\} \tag{1.3}$$

where D is the set of individuals in the pedigree whose phenotype with respect to the trait of interest has been observed, $\phi(D)$ is the set of phenotypes for those individuals, G is

the pedigree structure and M is the genetic model being considered for the trait. Varying the different parameters in Equation (1.3) allows us to calculate different likelihoods. For example, if we allow M to vary we obtain $L(M)$, which represents the likelihood for any particular genetic model, and will allow us to infer the mode of inheritance of the trait in question. Varying G results in $L(G)$, which represents the likelihood of any particular pedigree configuration, and can be used for inferences on the structure of the pedigree. We can use the conditional independence property to represent Equation (1.3) in terms of τ, ρ and π as defined in Definitions (1.2, 1.3, 1.4).

$$L\{\phi(D), G, M\} = \sum_{i_1} \cdots \sum_{i_n} \prod_{j \in F} \pi(i_j) \prod_{k \notin F} \tau(i_k, i_{f_k}, i_{m_k}) \prod_{l \in D} \rho\{\phi(l), i_l\}, \qquad (1.4)$$

where F is the set of founders of the pedigree, n is the number of individuals in the pedigree G, and f_j and m_j are the father and mother of individual j respectively.

1.4 Gene Counting

In this thesis we are concerned with ancestral inference and not interested in changing the genetic model; in addition, all of the traits are co-dominant. This means that both the transmission and penetrance probabilities are fixed and so in Equation (1.4) only the population allele frequencies are unknown.

Ceppellini, Siniscalco, and Smith (1955) illustrated a new method called *gene counting* to estimate the allele frequencies. They used recessive models where the phenotypes do not exclusively represent the genotypes, as we have shown an individual of blood group **a** may be genotypically AO or AA, but by grouping all the genotypes that express the given phenotype they were able to determine the gene frequencies by a simple allele count. This gives new improved estimates of the allele frequencies which provides us with the possibility of iterating to convergence. Smith (1957) generalised the procedure of estimation by counting and showed its applicability to a wide range of problems. Ceppellini, Siniscalco, and Smith (1955) indicated that this procedure led to maximum likelihood estimates (MLEs) and Smith (1957) postulated that these estimates from the counting methods were always the MLEs.

1.5 The EM Algorithm

However, it was not until Dempster, Laird, and Rubin (1977) that a rigorous approach was formulated for this method of MLEs. They observed that if the MLEs were difficult to obtain through the usual method of differentiation, then the obtained expression could be represented in an iterative form. A brief account of the algorithm is given below.

Introduction

The EM algorithm provides a general approach to iterative computation of ML estimation when the observations can be viewed as incomplete data. Each iteration consists of an Expectation step followed by a Maximisation step, hence the EM algorithm. The term *incomplete data* in its general form implies the existence of two sample spaces $\mathcal{Y}$ and $\mathcal{X}$ and a many - one mapping from $\mathcal{X} \rightarrow \mathcal{Y}$. The observed data $\mathbf{y}$ are a realisation from $\mathcal{Y}$. The corresponding $\mathbf{x} \in \mathcal{X}$ is not directly observed but indirectly through $\mathbf{y}$. More specifically, we assume there is a mapping $\mathbf{y} = \mathbf{y}(\mathbf{x})$ from $\mathcal{X}$ to $\mathcal{Y}$ and that $\mathbf{x}$ is known only to lie in $\mathcal{X}(\mathbf{y})$, the subset of $\mathcal{X}$ determined by the equation $\mathbf{y} = \mathbf{y}(\mathbf{x})$, where $\mathbf{y}$ is the observed data. The authors consider a family of sampling densities $f(\mathbf{x} \mid \phi)$ depending on parameters ϕ and derive its corresponding family of sampling densities $g(\mathbf{y} \mid \phi)$. The complete data specification $f(\ldots \mid \ldots)$ is related to the incomplete data specification $g(\ldots \mid \ldots)$ by

$$g(\mathbf{y} \mid \phi) = \int_{\mathcal{X}(\mathbf{y})} f(\mathbf{x} \mid \phi)\, d\mathbf{x}. \tag{1.5}$$

The EM algorithm is directed at finding a value of ϕ which maximises $g(\mathbf{y} \mid \phi)$ given an observed $\mathbf{y}$ by making use of the associated family $f(\mathbf{x} \mid \phi)$.

Definitions of the EM Algorithm

We now define the EM algorithm. Suppose that $f(\mathbf{x} \mid \phi)$ has the regular exponential family form

$$f(\mathbf{x} \mid \phi) = b(\mathbf{x}) \exp\{\phi \mathbf{t}(\mathbf{x})^T\}/a(\phi) \tag{1.6}$$

where ϕ denotes a $1 \times r$ vector parameters, $\mathbf{t}(\mathbf{x})$ denotes a $1 \times r$ vector of complete data sufficient statistics and the superscript T denotes matrix transpose. The term regular means here that Φ is restricted only to an r dimensional convex set Ω such that Equation (1.6) defines a density for all ϕ in Ω and the sample space $\mathcal{X}$ over which $f(\mathbf{x} \mid \phi) > 0$ is the same for all ϕ in Ω. We notice that if

$$\frac{\partial}{\partial \phi_i} \log(f) = 0, \tag{1.7}$$

then

$$t_i(x) - \frac{a'(\phi)}{a(\phi)} = 0. \tag{1.8}$$

Now

$$a(\phi) = \int b(x) \exp\{\phi \mathbf{t}(\mathbf{x})^T\} d\mathbf{x}. \tag{1.9}$$

Hence,

$$\frac{a'(\phi)}{a(\phi)} = \frac{\int b(x)t_i(x)\exp(\phi t)dx}{a(\phi)} \tag{1.10}$$

$$= E(t_i(x)). \tag{1.11}$$

Thus, if Equation (1.7) holds for $i = 1, \ldots, n$

$$\mathbf{t(x)} = E(\mathbf{t(x)}). \tag{1.12}$$

So, finding the MLE of ϕ for a given value of $\mathbf{t(x)}$ is equivalent to finding the ϕ for which $E(\mathbf{t(x)})$ equals the observed $\mathbf{t(x)}$.

The algorithm works as follows. Suppose that $\boldsymbol{\phi}^{(p)}$ denotes the current value of $\boldsymbol{\phi}$ after p cycles of the algorithm, the next cycle can be obtained by:

- (E–Step): Find

$$\mathbf{t}^{(p)} = E(\mathbf{t(x)} \mid \mathbf{y}, \boldsymbol{\phi}^{(p)}) \tag{1.13}$$

- (M–Step): Determine $\boldsymbol{\phi}^{(p+1)}$ as the solution of the equations

$$E(\mathbf{t(x)} \mid \boldsymbol{\phi}) = \mathbf{t}^{(p)}. \tag{1.14}$$

These two steps are then repeated until convergence.

Remarks.

- The E step can be thought of as estimating the *complete-data* set but doesn't necessarily correspond to any possible $\mathbf{x}$.

- As previously explained, Equation (1.14) is equivalent to ML estimation of $\boldsymbol{\phi}$ from $\mathbf{t(x)} = \mathbf{t}^{(p)}$.

- It is assumed that if $\mathbf{x}$ was observed then it would be straightforward to find the $\hat{\boldsymbol{\phi}}$ that maximises $f(\mathbf{x} \mid \boldsymbol{\phi})$ – or equivalently, solves Equation (1.14).

- The appeal of this algorithm is its simplicity and generality. For example at the estimation step, the sufficient statistics can be used rather than the whole data. The complete data can include parameters.

- The likelihood is forced to increase after each iteration. Under certain additional conditions it can be shown that the algorithm does what it is intended to do - find the MLE. See (Dempster, Laird, and Rubin 1977) for further details.

Example 1.3. MLEs: Consider the *ABO* blood type system and suppose the three alleles A B and O have frequency π_1, π_2 and π_3 respectively.

Genotype	Phenotypes	Genotype Counts	Phenotype Freqs	Phenotype Counts
AA	**a**	N_{AA}	$\pi_1^2 + 2\pi_1\pi_3$	n_a
AO		N_{AO}		
BB	**b**	N_{BB}	$\pi_2^2 + 2\pi_2\pi_3$	n_b
BO		N_{BO}		
AB	**ab**	N_{AB}	$2\pi_1\pi_2$	n_{ab}
OO	**o**	N_{OO}	π_3^2	n_o
				n

□

Consider the the likelihood for the incomplete data

$$L(\boldsymbol{\pi}, \boldsymbol{n}) = c(\pi_1^2 + 2\pi_1\pi_3)^{n_a}(\pi_2^2 + 2\pi_2\pi_3)^{n_b}(2\pi_1\pi_2)^{n_{ab}}(\pi_3^2)^{n_o}. \tag{1.15}$$

We cannot solve this by using Lagrangian multipliers, therefore we use EM. Suppose that we had observed the genotype counts, $N_{AA}, N_{AO} \ldots N_{OO}$, the likelihood for the complete data is

$$L(\boldsymbol{\pi}, N) = c(\pi_1^2)^{N_{AA}}(2\pi_1\pi_3)^{N_{AO}}(\pi_2^2)^{N_{BB}}(\pi_2\pi_3)^{N_{BO}}(2\pi_1\pi_2)^{N_{AB}}(\pi_3^2)^{N_{OO}}. \tag{1.16}$$

We can now use the Lagrangian technique using the constraint $\sum_i \pi_i = 1$ to get

$$\hat{\pi_1} = \frac{2N_{AA} + N_{AB} + N_{AO}}{2N}, \hat{\pi_2} = \frac{2N_{BB} + N_{AB} + N_{BO}}{2N}, \hat{\pi_3} = \frac{2N_{OO} + N_{AO} + N_{BO}}{2N}. \tag{1.17}$$

This is the M-step for given $N_{AA}, N_{AO} \ldots N_{OO}$ where $N = (N_{AA} + N_{AO} + N_{BB} + N_{BO} + N_{AB} + N_{OO})$. For the E-step we need to get estimates of $N_{AA}, N_{AO} \ldots N_{OO}$ from $n_a, \ldots, n_o$ and $\hat{\pi_1}, \hat{\pi_2}, \hat{\pi_3}$. We use the following equations.

$$N_{AA} = \frac{n_a\hat{\pi_1}^2}{\hat{\pi_1}^2 + 2\hat{\pi_1}\hat{\pi_3}} \qquad N_{AO} = \frac{n_a 2\hat{\pi_1}\hat{\pi_3}}{\hat{\pi_1}^2 + 2\hat{\pi_1}\hat{\pi_3}} \qquad N_{AB} = \frac{n_{ab}2\hat{\pi_1}\hat{\pi_2}}{2\hat{\pi_1}\hat{\pi_2}}$$
$$N_{BB} = \frac{n_b\hat{\pi_2}^2}{\hat{\pi_2}^2 + 2\hat{\pi_2}\hat{\pi_3}} \qquad N_{BO} = \frac{n_b 2\hat{\pi_2}\hat{\pi_3}}{\hat{\pi_2}^2 + 2\hat{\pi_2}\hat{\pi_3}} \qquad N_{OO} = \frac{n_o\hat{\pi_3}^2}{\hat{\pi_3}^2}. \tag{1.18}$$

Iterating these two steps allows us to converge to the MLEs of the allele frequencies.

The EM algorithm has many applications in genetics. Elston (1969) gives lots of iteration methods in connection with genetical problems. Thompson's discussion of Dempster, Laird, and Rubin (1977) highlights the gene counting method of estimating the allele fre-

quencies from phenotype frequencies. The method works as the underlying distribution is a multinomial distribution which satisfies the exponential family condition needed in the stated EM algorithm. However, most of the applications are for population frequencies rather than pedigrees. For information on pedigrees we need to be able to do a more complicated E step. There are two possible ways of obtaining statistics on the pedigree – one is an algorithm called *peeling* and the other is a simulation method. We now discuss these two methods more generally.

1.6 Graphical Models and Peeling

In this section we shall briefly outline two different ways of calculating probabilities on a pedigree of the nested summation type shown by Equation (1.4). A technique called *peeling* was developed in the late 1970s (Cannings, Thompson, and Skolnick 1978) and turns out to be similar to more general methods of calculating probabilities for graphical models. The natural conditional independencies of a pedigree make it possible to think of it as a graphical model. Both methods of calculation make use of the conditional independencies. We shall look at both methods in turn and then try to show the similar ideas and methods.

1.6.1 Basic Nested Calculation

Consider the problem of solving a calculation of the form

$$p(x) = \sum_{x_1} \cdots \sum_{x_n} f(x_1, \ldots, x_n) \tag{1.19}$$

when $f(x_1, \ldots, x_n)$ factorises into functions of subsets of the variables $x_1, \ldots, x_n$.

$$\sum_{x_1} \cdots \sum_{x_n} f(x_1, \ldots, x_n) = \sum_{x_1} \cdots \sum_{x_n} \prod_{c \in \mathcal{C}} \phi_c(x) \tag{1.20}$$

where $\phi_c(x)$ depends only on $\{x_i; i \in c\}$.

The basic step is to write Equation(1.19) as

$$\sum_{x_1} \cdots \sum_{x_{k-1}} \sum_{x_{k+1}} \cdots \sum_{x_n} \left(\prod_{c \in \mathcal{C}_1} \phi_c(x)\right) \sum_{x_k} \left(\prod_{c \in \mathcal{C}_2} \phi_c(x)\right) \tag{1.21}$$

where $\mathcal{C}_1$ is the set of elements of $\mathcal{C}$ which do not contain k and $\mathcal{C}_2$ is the set of elements of $\mathcal{C}$ which does contain k.

We can write this as

$$\sum_{x_1} \cdots \sum_{x_{k-1}} \sum_{x_{k+1}} \cdots \sum_{x_n} \left(\prod_{c \in \mathcal{C}_1} \phi_c(x)\right) \phi_d(x) \tag{1.22}$$

where d is the set containing all indices $j \neq k$ which appear in $c \in \mathcal{C}_2$ and $\phi_d(x)$ depends on x_k and $\{x_j;\ j \neq k$ and $j \in c$ and $k \in c$ for some $c \in \mathcal{C}\}$.

However, by redefining $\mathcal{C}'$ as $\mathcal{C}_1 \cup \{d\}$ we get

$$\sum_{x_1} \cdots \sum_{x_{k-1}} \sum_{x_{k+1}} \cdots \sum_{x_n} \prod_{c \in \mathcal{C}'} \phi_c(x). \tag{1.23}$$

This step has the effect of summing out one of the variables but the remaining expression is of the same form as the original equation and this allows us to iterate until we have the answer.

Example 1.4. SUMMING OUT VARIABLES:

$$\begin{aligned} \sum_{x_1} \cdots \sum_{x_4} f(x_1, \ldots, x_4) &= \sum_{x_1} \sum_{x_2} \sum_{x_3} \sum_{x_4} \phi_1(x_1, x_2)\phi_2(x_1, x_3)\phi_3(x_1, x_4)\phi_4(x_2, x_3) \\ &= \sum_{x_1} \sum_{x_3} \sum_{x_4} \phi_2(x_1, x_3)\phi_3(x_1, x_4) \sum_{x_2} \phi_1(x_1, x_2)\phi_4(x_2, x_3) \\ &= \sum_{x_1} \sum_{x_3} \sum_{x_4} \phi_2(x_1, x_3)\phi_3(x_1, x_4)\phi_5(x_1, x_3). \end{aligned} \tag{1.24}$$

□

We look at two ways of calculating this sort of nested summation.

1.6.2 Computations on Graphical Models

We briefly review the field of graphical modelling but for a more in depth account see Whittaker (1990). Every new research area generates its own definitions and there follow some of the key definitions for graphical models which provide a way of calculating Equation (1.20).

Definitions

Definition 1.5. A *graph* G is a mathematical object that consists of two sets, a set of nodes K and a set of edges E, consisting of pairs of elements from K. We usually take K to be the set of natural numbers $\{1, \ldots, k\}$.

Definition 1.6. A *complete* graph is one in which every pair of nodes is joined by an edge.

Definition 1.7. A *path* is a sequence of distinct nodes $i_1, \ldots, i_m$ for which (i_l, i_{l+1}) is in E for each $l = 1, \ldots, m-1$.

Definition 1.8. A *cycle* is a path with $i_1 = i_m$.

Definition 1.9. A *chord* is a one edge shortcut across a path.

Definition 1.10. Let $a \subseteq K$ denote a subset of nodes of the graph. The set of *neighbours* of a, written $\delta(a)$, is the set of all the nodes, b, in K, but not in a, such that there is an edge from b to at least one node in a.

Definition 1.11. A *sub-graph* induced by the subset of nodes a, denoted G_a, is the graph obtained by deleting all the nodes not in a from the graph on K, together with all edges that do not join two elements of a.

Definition 1.12. A *clique* is a subset of nodes that induces a complete subgraph.

Definition 1.13. A clique is *maximal* if it is not a strict subset of another clique.

Example 1.5. SIMPLE GRAPH: Let's consider the simple graph shown in Figure 1-5 in order to explain some of the graphical definitions.

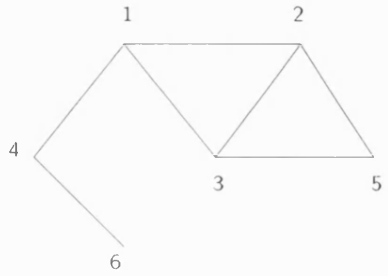


Figure 1-5: EXAMPLE OF A GRAPH: The figure shows a simple graph.

In this graph we have the following

Cliques:	{1}{2}{3}{4}{5}{6}
	{1,2}{1,3}{1,4}{2,3}{2,5}{3,5}{4,6}
	{1,2,3} {2,3,5}
Maximal Cliques:	{1,2,3}{2,3,5}{1,4}{4,6}
A Path:	{1,2,5,3} with a **Chord** (1,3)
A Cycle:	{1,2,5,3,1} with a **Chord** (2,3).

□

Triangulation

Definition 1.14. A *triangulated graph* is a graph in which any cycle of length 4 or more has a chord.

Triangulating a graph consists of adding new edges, *fill-ins*, so that the resulting graph is a triangulation.

Definition 1.15. A *minimal triangulation* is one in which no subset of fill-ins results in a triangulation.

Examples of triangulation and minimal triangulation are shown in Figure 1-6.

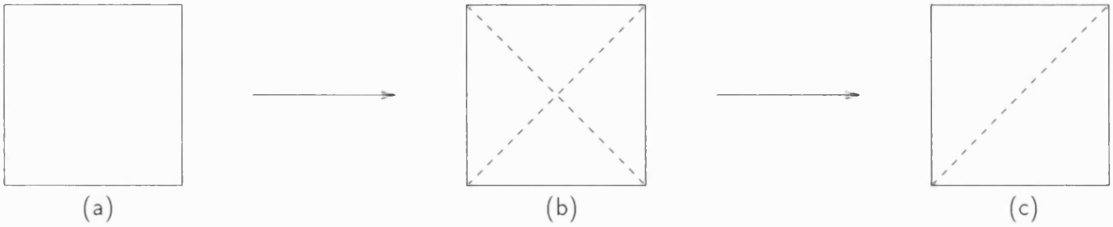


Figure 1-6: MINIMAL TRIANGULATION OF A GRAPH: Figure (a) shows a simple untriangulated graph. Figure (b) shows a possible triangulation and Figure (c) shows a minimal triangulation as we can remove one of the diagonals and the graph is still triangulated.

Definition 1.16. An *extremal* node i is one for which $\delta(i)$ is complete *i.e.*, $\delta(i)$ is a clique.

We shall use graphs to illustrate the structure of a multivariate probability distribution. Suppose $X = (X_1, \ldots, X_n)$ and

$$f(x) = \prod_{c \in \mathcal{C}} \phi_c(x) \tag{1.25}$$

where $\phi_c(x)$ depends only on $\{x_i;\ i \in c\}$.

Definition 1.17. A *moral graph* for Equation (1.25) consists of a set of nodes $1, \ldots, n$ with i and j joined $\iff$ i and $j \in c$ for some $c \in \mathcal{C}$.

We note that $c \in \mathcal{C}$ defines cliques in the graph but not necessarily maximal cliques, or all of the cliques. The moral graph provides a pictorial representation of the structure of $f(x) = \prod_{c \in \mathcal{C}} \phi_c(x)$.

Example 1.6. MORAL GRAPHS: Suppose we have a function

$$f(x_1, \ldots, x_4) = \phi_1(x_1, x_2)\phi_2(x_1, x_3)\phi_3(x_1, x_4)\phi_4(x_2, x_3) \tag{1.26}$$

this can be represented as the moral graph shown in Figure 1-7.

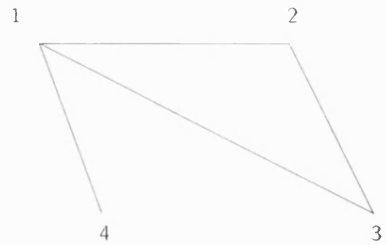


Figure 1-7: EXAMPLE OF A MORAL GRAPH: The figure corresponds to the moral graph of the function given in Equation (1.26).

□

If we refer to Equation (1.24), when we sum out a variable, x_2, we involve all other variables that are arguments of a function for which x_2 is an argument, in this case

x_1 and x_3. In general, in summing out variable x_i, we involve all variables corresponding to nodes connected to i in the moral graph, *i.e.*, $\delta(i)$.

If we sum out the variables in the order x_4, x_1, x_3, x_2 we obtain the sequence of equations given in Equations (1.27) and the sequence of moral graphs shown in Figure 1-8.

$$\begin{aligned}
&\sum_{x_1}\sum_{x_2}\sum_{x_3}\sum_{x_4}\phi_1(x_1,x_2)\phi_2(x_1,x_3)\phi_3(x_1,x_4)\phi_4(x_2,x_3) \\
&\qquad = \sum_{x_1}\sum_{x_2}\sum_{x_3}\phi_1(x_1,x_2)\phi_2(x_1,x_3)\phi_4(x_2,x_3)\phi_5(x_1) \\
&\qquad = \sum_{x_2}\sum_{x_3}\phi_4(x_2,x_3)\phi_6(x_2,x_3) \\
&\qquad = \sum_{x_2}\phi_7(x_2).
\end{aligned} \tag{1.27}$$

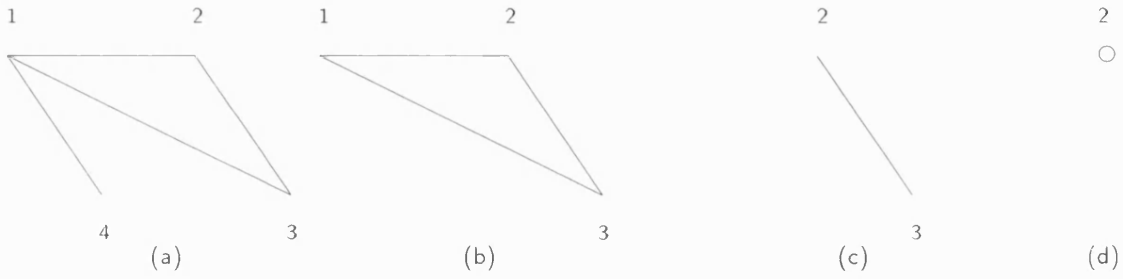


Figure 1-8: SEQUENCE OF MORAL GRAPHS: Figures (a) - (d) represent the moral graphs obtained after summing out variables x_4, x_1, x_3, x_2 respectively from Equation (1.26).

A systematic approach to doing nested sums is as follows.

1. Draw the moral graph.

2. Triangulate.

3. Sum out one extremal node at a time.

The reason for summing an extremal node at each stage is that then the sequence of moral graphs is just the sequence of sub-graphs of the remaining nodes. This can be proved using Dirac's Theorem which is stated below without proof.

Theorem. DIRAC: *A triangulated graph is either complete or contains at least 2 extremal nodes contained in two distinct maximal cliques. Furthermore, if i is an extremal node of a triangulated graph, then the sub-group induced by all other nodes is also triangulated.*

Proof. See Dirac (1961). □

Efficient Calculation

Example 1.7. GRAPH TRIANGULATION: Let's consider the problem of summing

$$f(x_1,\ldots,x_5) = \phi_1(x_1,x_2)\phi_2(x_2,x_3)\phi_3(x_3,x_4)\phi_4(x_1,x_4)\phi_5(x_2,x_3,x_5) \tag{1.28}$$

over all values of $x_1,\ldots,x_5$. The graph that corresponds to this function is given in Figure 1-9(a).

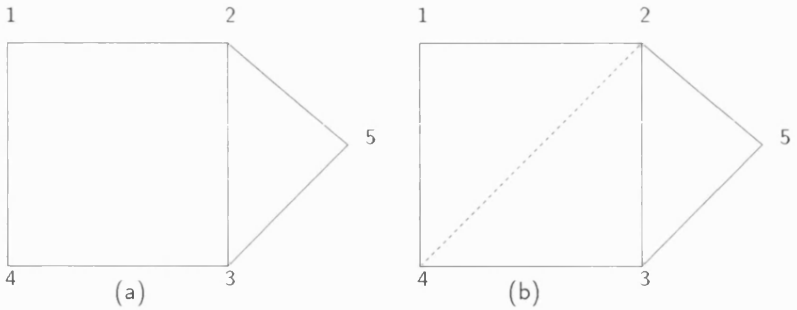


Figure 1-9: EXAMPLE OF A TRIANGULATION: Figure (a) represents the graphical model of Equation (1.28). Figure (b) represents the original graph with the fill-in (2,4) that makes the graph triangular.

□

The summation connected with Example 1.7 is

$$\begin{aligned}&\sum_{x_1=1}^{n_1}\sum_{x_2=1}^{n_2}\sum_{x_3=1}^{n_3}\sum_{x_4=1}^{n_4}\sum_{x_5=1}^{n_5}\phi_1(x_1,x_2)\phi_2(x_2,x_3)\phi_3(x_3,x_4)\phi_4(x_1,x_4)\phi_5(x_2,x_3,x_5)\\ &\quad=\sum_{x_1=1}^{n_1}\sum_{x_2=1}^{n_2}\sum_{x_3=1}^{n_3}\sum_{x_4=1}^{n_4}\phi_1(x_1,x_2)\phi_2(x_2,x_3)\phi_3(x_3,x_4)\phi_4(x_1,x_4)\phi_6(x_2,x_3).\end{aligned} \tag{1.29}$$

Finding ϕ_6 for each (x_2,x_3) pair needs $n_2 \times n_3$ cases to be evaluated and summing $\phi_5(x_2,x_3,x_5)$ needs n_5 values of x_5; so let us call the total calculation cost $n_2 \times n_3 \times n_5$. Similarly, the following iterative steps in the calculation need

SUMMATION	COST
$\sum_{x_1}\sum_{x_2}\sum_{x_4}$	$O(n_2 \times n_4 \times n_3)$
$\sum_{x_1}\sum_{x_2}$	$O(n_1 \times n_2 \times n_4)$
$\sum_{x_1}$	$O(n_1 \times n_2)$
	$O(n_1)$.

Now if we simplify to the case where $n_1,\ldots,n_5 = h$, then the total computational cost must be $< O(5h^3)$. More generally, when summing out over x_i which starts off in a clique

of size k say, calculations are of $O(h^q)$ where q is the number of neighbours of x_i at this point and we note that $q < k$ if others in the clique have already been dealt with. This implies that the largest computational cost is $O(h^m)$ where m = size of the biggest clique in the triangulated graph. This leads to a total cost $< O(nh^m)$ overall.

We have shown that we have to triangulate the graph and would like to use a minimal triangulation. In a large and complicated graph there could be many choices and for computational optimality would like to aim for the minimum size of the maximum clique. For this we need to use some optimisation algorithm, such as simulated annealing – see §1.12.3.

1.6.3 Peeling

The basis of peeling is to remove individuals recursively from a pedigree, passing the information held on them onto a subset of the remaining individuals, termed a *cutset*. Elston and Stewart (1971) introduced a method for computing likelihood for simple pedigrees. The method iteratively peels up each generation onto the generation above to give a likelihood on the whole pedigree.

Ott (1974) extended this by deriving a conditional probability expression for the case where one parent has parents who are both founders, so allowing for probability calculations to be formed on a wider class of pedigrees.

Extensions to this method were made by Lange and Elston (1975) and enabled calculations on looped pedigrees. Their method identified marriages in a loop, the loop next to one of the partners of the marriage was severed and the severed individual's genetic material was duplicated. This procedure was repeated until it produced a set of simple pedigrees. The overall likelihood is then calculated by conditioning for every possible value on each duplicated individual, calculating the results on each simple pedigree and combining the results. The method works in principle for any pedigree but in reality flounders for large complex pedigrees due to large number of breaks that are needed, producing a prohibitively large number of simple pedigrees. This method was suggested by Pearl (1986) to deal with a similar problem in expert systems.

Cannings, Skolnick, and Thompson (1976) generalised the method to allow the algorithm to be used for zero-loop pedigrees by collapsing the information downwards from parents onto children. The algorithm was based upon each individual in the pedigree partitioning the pedigree into an upper and lower section *i.e.*, each individual acting as a separator giving conditional independence between the sections. Using four peeling operations: offspring onto parent pair, summing over peripheral pairs, combining information from upper sections of parents and peeling downwards from parents to offspring, they show that this is sufficient to peel any zero-loop pedigree.

They extended this to arbitrarily complex pedigrees by replacing the notion of upper and lower sections, which have little meaning for single individuals in a graph with loops, by choosing a sequence of cutsets of individuals, each cutset partitioning the pedigree into two sub-pedigrees. The basis of peeling is that knowledge of the cutset provides conditional

independence between the peeled and unpeeled section of the graph, each cutset splitting the pedigree into two sections, the peeled set consisting of individuals whose information has been incorporated into the calculation, called the *R-function* by the authors, on the cutset and the unpeeled set. The *R-function* is defined to be the probability distribution of the individuals on the cutset conditional on the set of peeled individuals. That is for each cutset in turn we remove a peeled set of individuals from the graph, expressing the information contained on the peeled vertices on the cutset. Doing this repeatedly allows us to move through the pedigree, expressing all of the information contained on those individuals so far peeled as a function of a small group of individuals. So the peeled set, initially empty, grows monotonically as it extends to cover the whole graph, when the unpeeled set is either empty or a set of individuals for which we will have calculated probabilities. The information from every individual is incorporated exactly once.

Optimisation

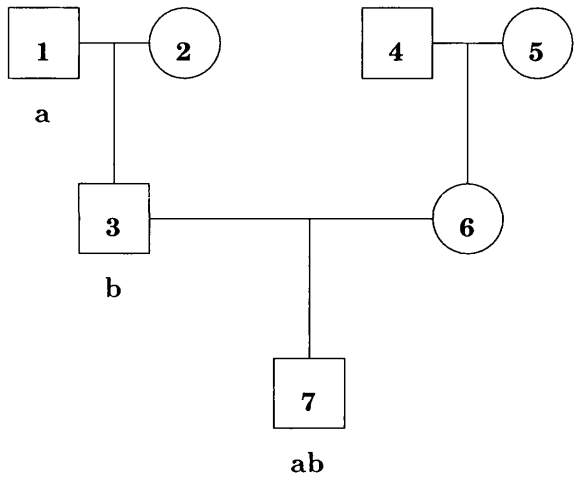


Figure 1-10: ABO BLOOD-GROUP SYSTEM: The figure shows a pedigree in which three of the individuals have been typed for a phenotype from the *ABO* Blood Group system.

Unfortunately, this valuable tool for finding exact probabilities on complex pedigrees has its limitations and the problem is to do with finding the sequence of cutsets. Thomas (1985) indicates that if we consider a genetic trait with k genotypes then for a cutset of n individuals, the storage requirement is $O(k^n)$. For example, depending on the complexity of the pedigree, a cutset of 13 is sufficient to cause computational problems in the case of a diallelic trait with three possible genotypes and for a multi-allelic system problems can arise with smaller cutsets. However, Thomas (1985) shows that an ordering of marriages in a pedigree induces a sequence of cutsets and investigates various minimisation algorithms, see §1.12.3 and §2.2, to try and find sequences which will reduce the computational requirements.

Example 1.8. MARRIAGE ORDERING EFFECT ON THE CUTSETS: Figure 1-10 represents a pedigree with three individuals, 1, 3 and 7, assigned a phenotype of the *ABO* blood group system. Equation(1.4) gives us

$$L\{\phi(D), G, M\} = \sum_{x_1} \cdots \sum_{x_7} \pi(x_1)\pi(x_2)\pi(x_4)\pi(x_5)\tau(x_3|x_1x_2)\tau(x_6|x_4x_5)\tau(x_7|x_3x_6) \cdot \rho(\mathbf{a}|x_1)\rho(\mathbf{b}|x_3)\rho(\mathbf{ab}|x_7)$$

$$= \sum_{x_5}\sum_{x_4} \pi(x_4)\pi(x_5) \sum_{x_6} \tau(x_6|x_4x_5) \sum_{x_3}\sum_{x_7} \rho(\mathbf{ab}|x_7)\rho(\mathbf{b}|x_3)\tau(x_7|x_3x_6) \cdot \sum_{x_2}\sum_{x_1} \tau(x_3|x_1x_2)\rho(\mathbf{a}|x_1)\pi(x_1)\pi(x_2). \tag{1.30}$$

If the calculation is summed out in the order $7,6,5,4,3,2,1$, then the time required is $O(g^7)$, whilst summing out in the order $1,2,7,3,6,4,5$ reduces the time to $O(g^3)$, where g is defined to be the number of values taken by each x_i. More importantly, it also decreases the storage requirement from $O(g^6)$ to $O(g^1)$. □

Similarity to Graphical Models

We can represent the pedigree format introduced in §1.2.2 now as a form of a moral graph and use graphical methods to calculate probabilities on the pedigree. We remember that the form of function to be evaluated by peeling is given by

$$P\{\phi(D)|G, M\} = \sum_{i_1} \cdots \sum_{i_n} \prod_{j\in F} \pi(i_j) \prod_{k\notin F} \tau(i_k, i_{f_k}, i_{m_k}) \prod_{l\in D} \rho\{\phi(l), i_l\}, \tag{1.31}$$

which is just the sort of problem for the triangulation method. Note that $\pi(i_j)$ is a function of one variable and $\rho\{\phi(l), i_l\}$ is a function of one variable when $\phi(l)$ is observed. So the cliques arise from the term $\prod_{k\notin F} \tau(i_k, i_{f_k}, i_{m_k})$ and the moral graph is obtained by joining each individual to its parents, offspring and spouses; this can be obtained from the marriage node graph by adding an edge connecting each pair of parents. An example of a simple pedigree and its corresponding moral graph is given in Figure 1-11.

The corresponding moral graph is triangulated and can be peeled using the sequence $x_3, x_1, x_2, x_8, x_7, x_6, x_5, x_4$. If the pedigree has no loops, *i.e.*, no in-breeding, then it can be peeled in $O(ng^3)$ time, where n = is the number of individuals, however if there is inbreeding then fill-ins might be required, sharply increasing computational costs.

Despite different terminology, the methods are essentially the same. Computations depend on the order in which variables x_i are summed out and efficiency is achieved by optimising the sequence of cutsets or by triangulating the graph which are equivalent.

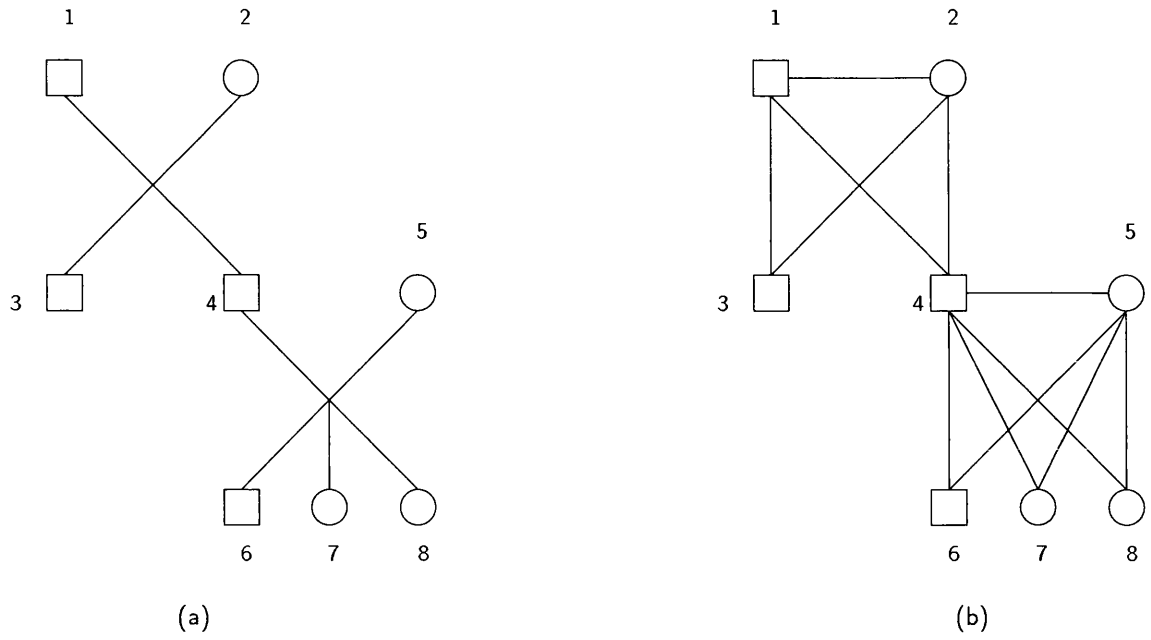


Figure 1-11: PEELING TRIANGULATION OF A GRAPH: Figure (a) shows a simple pedigree and Figure (b) shows the corresponding moral graph.

Expert Systems

An *expert system* is a computer program which is used to provide help for users by analysing complex situations and relationships based upon simpler rules and relationships that build up to form a more complicated and involved system. These rules given to it from a knowledge base which is typically derived from a combination of both subjective and objective opinion. The overall structure is assumed to hinge on a series of local relationships between the entities. There are a variety of philosophies on the development and representations of expert systems but the one that we are interested in here is the probabilistic approach where relationships are given in terms of probabilistic dependencies. This gives rise to graphs termed *belief networks* or *Bayesian networks* and when the nature of the dependencies between the two variables is perceived as being causal, not just associative, a *causal network*.

To specify such a system we need a vector of variables or entities $V = (v_1, \ldots, v_n)$ and in order to express the relationships between these variables, a conditional probability statement for each variable expressing its probability distribution in terms of parent variables, which might possibly be empty. The graphical structure is then the directed graph $g = (V, E)$ where V is the set of random variables and $(v_i, v_j) \in E$ if v_i is a parent of v_j.

Lauritzen and Spiegelhalter (1988) draw a firm link between genetics and expert systems proving that the procedures described in the above paper and Cannings, Thompson, and Skolnick (1978) are in fact equivalent. Although they point out that the expert systems algorithm allows for calculation of marginals on all nodes throughout the structure

at the expense of higher storage requirements. This is because the functions on the cliques not just the clique separators are stored; it is also better designed for receiving information in batches, neither of which is particularly important in a genetic context. Lauritzen and Spiegelhalter introduce the idea of displaying the pedigree in the style of a moral graph moving away from the the more structured, time-order approach that had been previously used. This representation with its associated evidence potential structure allows for easier incorporation of new genotype or other types of information into the pedigree, simply by adding a new potential function on the relevant individuals.

Conclusion

The peeling technique and equivalent computational methods for Graphical Models gives us a powerful method for finding exact probabilities on pedigrees but this method is computationally infeasible on complex pedigrees due to the vast amounts of memory required. Even using an optimisation algorithm like simulated annealing to reorder the summing out of individuals and form smaller cutsets, doesn't reduce the memory requirements on the Przewalski Horse pedigree sufficiently for a four allele trait. So, interest has moved on to simulation methods to see if they provide further improvements.

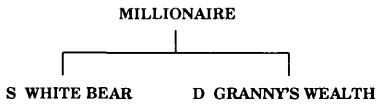
1.7 Simulation

The technique of simulation is not a modern idea in pedigree analysis. In this section we briefly outline one of the earliest methods and a more recent computational method which has had some success.

1.7.1 Wright and McPhee

Wright and McPhee (1925) used random sampling of ancestral lines to estimate a pedigree statistic of interest. To achieve truly random lines for an individual a simulated two column pedigree was created, one column for the sire and one column for the dam. A coin toss was used for each column to decide which parental line in the previous generation should be followed: a head resulted in the sire line being followed and a tail the dam line. For each run of the simulation two alternatives were possible, either there would be no individual who appeared in both columns of this new simulated pedigree and so no common ancestor was observed, or an individual would be in both columns and so would be a common ancestor. An example of a simulation from Wright and McPhee (1925) which shows common ancestry is shown in Figure 1-12.

Even with only 1024 simulations, good approximations are obtained by the method. An approximation of a pedigree statistic for Millionaire, shown in Figure 1-12, of 0.19 ± 0.0052 compares well with the exact result 0.192. However, when we say only 1024 simulations we must remember how much effort went into generating *Bernoulli* random variables thousands and thousands of times in 1925.



D AMETHYST
D ARTLESS
D AMY
S GRAND DUKE 36TH
S GRAND DUKE 30TH
S 3RD DUKE OF CLARENCE
D DUCHESS 80TH
D DUCHESS 72ND
D DUCHESS 67TH
D DUCHESS 59TH
S 2ND DUKE OF OXFORD
D OXFORD 2ND
S SHORT TAIL
S BELEVEDERE
D ANGELINA
S YOUNG WYNYARD
S WELLINGTON
S COMET
* S FAVOURITE
S BOLINGBROKE
D YOUNG STRAWBERRY
* D FAVOURITE COW
D UNKNOWN
S SMITH'S BULL

S SIGN OF RICHES
S LORD MAYOR
D LADY MYSIE
S FIELD MARSHAL
S ROAN GAUNLET
D PRINCESS ROYAL
S CHAMPION OF ENGLAND
D VIRTUE
S PLANTAGENET
S DUKE OF LANCASTER
S DUKE
D HARTFORTH CHERRY
S MAJOR BOWN
S CHARLES II
D CORA
D MARIA
D GAUDY
* S FAVOURITE
D PHOENIX
* D FAVOURITE COW
S ALCOCK'S BULL
S UNKNOWN

Figure 1-12: WRIGHT'S SIMULATION: An example of a two line simulation of the Shorthorn pedigree. The initial before the animal's name relates to whether the (S)ire or (D)am line was taken and * represent a common ancestor (from Wright and McPhee (1925)).

1.7.2 Gene Drop

MacCluer, Vandeburg, Read, and Ryder (1986) introduced an important technique which enabled probabilities to be estimated for a wide variety of purposes in pedigree analysis. The paper cites the extent of genetic variability, predicting the risk of further loss of genes contributed by the various founders and illustrates some of these on several animal pedigrees.

The most attractive thing about the method is that it is easily understandable and can be easily implemented in the form of a computer program. The method allows for uncertainty as standard errors can be easily found for the required estimate.

Algorithm 1.1. GENE DROP: *For the procedure, it is assumed that the population is diploid, i.e., two alleles at a locus.*

- *Assign two uniquely labelled genes to each founding member of the population.*
- *Drop these genes to successive generations by using a random number generator to simulate a sequence of Bernoulli trials with probability* $\frac{1}{2}$ *in order to decide which parental gene is inherited.*
- *At the end of a single gene drop, all members of the pedigree have had their genes assigned.*
- *Various counts are made depending on the required functional estimate.*
- *The process is repeated thousands of times to obtain good estimates.*

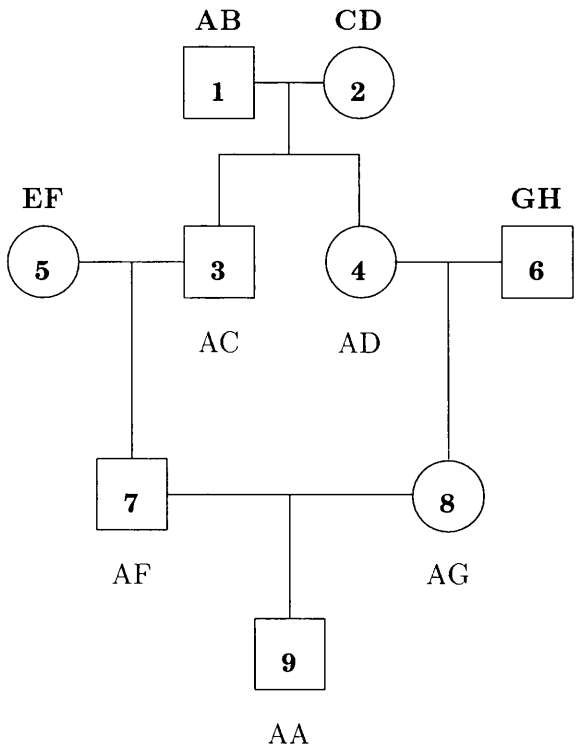


Figure 1-13: GENE DROP: Shows a possible realisation from a gene drop simulation for the example pedigree given in Figure 1-3. Two unique alleles, shown in bold, are assigned to each founder and then dropped to descendants through the remainder of the pedigree according to Mendelian transmission.

□

One possible outcome of a gene drop simulation is shown in Figure 1-13. The method is very successful when there are no observed phenotypes and the interest is in finding probabilities that are concerned with the structure of the pedigree. In this case, all simulations are allowed as there is no rejection, and the method is an efficient way of obtaining approximations. Thomas (1990) addresses the problem of wanting to know how close the simulation estimates are to the real probabilities. For numerous animal pedigrees with no observed data, he calculates exact probabilities via the peeling algorithm, §1.6, and compares them with the estimates obtained from the gene drop method. He calculates the number of simulations required to be within a certain confidence interval and also gives the computational time for each method; he concludes that the gene dropping method yields very good approximations, which may be improved by antithetic variance methods, but may be over-used for simple pedigrees for which exact results are available.

However, if there are any observed data on the pedigree then this method begins to falter as it involves rejecting simulated genotype configurations that are inconsistent with the data. As the amount of observed data increases, as for example in the Przewalski Horse pedigree where 147 out of the 244 horses have been tested for phenotypes, the rejection rate becomes impractically high.

A new method of simulation called Markov Chain Monte Carlo (MCMC) has been recently developed and may be used to overcome this problem. It uses a Bayesian framework

which we briefly outline.

1.8 The Bayesian Model

We introduce the ideas needed to define the Bayesian model.

Prior

The true representation, denoted by $X = \{X_1, X_2, \ldots, X_n\}$, is treated as a random variable. We use $\pi(x)$ for the density of this random variable, where $x \in \Omega$ and Ω is defined to be the set of all possible configurations. The random variable X_i is some item of interest of an individual; in pedigree analysis this is the genotype, phenotype or some model parameters.

The key to reducing the computational cost of highly structured stochastic systems is the concept of conditional independence where each variable is related conditionally or locally to only a few other variables. We use this local conditional structure to model the true scene economically and this allows the model to exhibit global complexity even though the model has a simple local structure. We define the related variables using a Markov Random Field (MRF).

Markov Random Field

The idea of a MRF relies on the following definitions.

Definition 1.18. We say that two elements are neighbours if they are, in some sense, near each other. Denote a *neighbourhood* relation on $i, j \in X$ by $i \sim j, i, j \in \{1, \ldots, n\}$.

The relation $\sim$ is symmetric, so $i \sim j \Rightarrow j \sim i$. A standard condition is that i is not a neighbour of itself, $i \not\sim i$. The implication of this definition is that the neighbourhood structure is quite local but only if the neighbour relation is appropriately defined.

Definition 1.19. We let $\partial(i)$ represent the set of neighbours of x_i.

Definition 1.20. A *clique* is either a set of indices of elements, all of whom are neighbours, or a single index. Define C to be the set of all these cliques. Define, for each $i = 1, \ldots, n$ X_{-i} to be the set of all elements of X excluding element X_i, $X_{-i} = \{X_j \in X : j \neq i; j = 1, \ldots, n\}$.

Model 1.1. MARKOV RANDOM FIELD: *A MRF is defined by two conditions. Firstly, we require a positivity condition to ensure that every permitted configuration is considered.*

$$\Pr(X = x) > 0, \quad \forall\, x \in \Omega. \tag{1.32a}$$

Secondly, if we allow stochastic interactions between each individual i, and the defined set of neighbours, δ_i, then

$$\Pr(X_i = x_i \mid X_j = x'_j, \forall j \neq i) = \Pr(X_i = x_i \mid X_j = x'_j, \forall j \in \partial(i)). \tag{1.32b}$$

A theorem due to Hammersley and Clifford provides the general form for the probability density function (p.d.f.) of a MRF.

Theorem. HAMMERSLEY–CLIFFORD: *The general form of the p.d.f. that satisfies the two conditions for a MRF, in Equations* (1.32), *is*

$$f_X(x) = \Pr(X = x) = k \exp\{-E_X(x)\}, \quad \text{for } x \in \Omega \tag{1.33}$$

where $E_X(x) = \sum_{c \in C} E_c(x)$ and $E_c(x)$ depends only on the elements in the clique c. The constant k ensures that f_X integrates to one but its value need not be known to implement the method.

Proof. See Besag (1974) and Besag (1986). □

The term $E_X(x)$ is sometimes called the *energy function* (or *cost function*) of the configuration x.

Random field models are derived historically from models in statistical physics. Developments in the model and the design of algorithms that use this model have proceeded in parallel. A MRF can be summarised succinctly as a non-directional, non-causal, conditionally independent structure. Note that in this formula, each x_i will appears in only a small number of terms with some x_js. Later on we will see how this reduces computational cost.

Likelihood

Let the probability density of the observed configuration X_{obs} given $X_{unobs} = x_{unobs}$ be $f_{X_{obs}|X_{unobs}}(X_{obs}, x_{unobs})$ and let $f_{X_{obs}}(X_{obs})$ denote the marginal density of X_{obs}. The density $f_{X_{obs}|X_{unobs}}(X_{obs} \mid X_{unobs})$ is the likelihood distribution and although X_{obs} is a random variable, it is fixed by observation once the data have been observed.

Posterior

Let the probability density of the true pedigree given X_{unobs} be $f_{X_{unobs}|X_{obs}}(x_{unobs} \mid x_{obs})$. This density is the *posterior* distribution. In Bayesian analysis, inference about X is based on the posterior density, which in this case is the conditional distribution $X_{unobs} \mid X_{obs} = x_{obs}$.

$$f_{X_{unobs}|X_{obs}}(x_{unobs}, x_{obs}) = f_{X_{obs}|X_{unobs}}(X_{obs}, x_{unobs}) f_{X_{unobs}}(x_{unobs}) / f_{X_{obs}}(X_{obs}) \tag{1.34}$$

$$\propto f_{X_{obs}|X_{unobs}}(X_{obs}, x_{unobs}) f_{X_{unobs}}(x_{unobs}), \qquad x \in \Omega. \tag{1.35}$$

The posterior distribution is obtained by combining the prior information with the likelihood. Due to the high dimensionality of x, standard analytical, numerical or simulation methods may not be adequate. We are interested in simulating properties of the posterior such as expectations, variances or modes over some defined multivariate distribution or some function of a marginal posterior distribution. So for example, the functional of interest might be probabilities or vectors of probabilities.

The remaining problem is how to carry out computations with a MRF model. We wish to sample from the posterior distribution. Ideally, we require a sequence of independent realisations but this is generally impossible. The best we can do is to simulate from a Markov Chain where realisations are going to be dependent but only on the last realisation. The technique we use is called Markov Chain Monte Carlo (MCMC) but before launching into MCMC simulation we need to review the theory of Markov Chains. For a fuller account see an elementary book on probability such as Feller (1968).

1.9 Markov Chains

Definition 1.21. A stochastic process $\{X^t : t = 1, 2, \ldots\}$, is said to have the *Markov property* if future events depend on the current status and not the past. The discrete case is expressed as

$$\Pr(X^{t+s} = x^{t+s} \mid X^0 = x^0, \ldots, X^t = x^t) = \Pr(X^{t+s} = x^{t+s} \mid X^t = x^t) \quad \forall \quad s > 0 \quad \forall x^0, \ldots, x^t, x^{t+s}. \tag{1.36}$$

Definition 1.22. A *Markov Chain*, M, is a discrete time, time homogeneous, Markov process, with a countable state space that satisfies Equation (1.36).

In this case

$$\Pr(X^{t+1} = x' \mid X^t = x) = \Pr(X^1 = x' \mid X^0 = x) \tag{1.37}$$

where time $t \in \mathcal{N}$, and x and x' are possible states.

Definition 1.23. The Markov process can be defined by one-step *transition* probabilities

$$q(x - x') = \Pr(X^{t+1} = x' \mid X^t = x) \tag{1.38}$$

which can be thought of as the probability that you move to state x' given that you are in state x.

These transition probabilities can be collected together into a *transition matrix*, $Q = \{q(x - x') : x - x' \in \Omega\}$.

Definition 1.24. A Markov Chain is defined as *irreducible* if we can reach every state from every other.

Definition 1.25. The *period* D_x of state x is defined as

$$D_x = \text{g.c.d.}\{t : \Pr(\text{getting from } x \text{ to } x \text{ in } t \text{ steps}) > 0, \text{for } t \in \mathcal{N}\}.$$

State x is *periodic* if $D_x > 1$.

Definition 1.26. A Markov Chain is *aperiodic* if it has a period of 1. An irreducible Markov Chain is aperiodic if it has a state x of period 1 and we only need to show that one state is aperiodic to ensure that the whole Markov Chain is aperiodic.

Definition 1.27. For a p.d.f. π, the *detailed balance condition* is satisfied iff

$$\pi(x)Q(x \to x') = \pi(x')Q(x' \to x) \; \forall \, x, x' \in \Omega, \tag{1.39}$$

where Q is a transition matrix that specifies an irreducible, aperiodic Markov Chain. The *general balance* condition is satisfied iff

$$\sum_{x'} \pi(x') \, Q(x' \to x) = \pi(x). \tag{1.40}$$

The detailed balance condition is also called *local balance* or *time reversibility.* Detailed balance implies general balance and it is often easier to work with.

The theorem which drives Markov Chain sampling is now given, a proof of which can be found in Feller (1968). It shows that under certain conditions the Markov Chain will converge to a unique stationary distribution.

Theorem. ERGODICITY: *If* $\{X^t\}$ *is an aperiodic, irreducible, finite Markov Chain, there exists a unique stationary distribution* $\pi(x)$ *satisfying*

$$\sum_{x'} \pi(x') \, Q(x' \to x) = \pi(x) \quad \forall x \in \Omega, \quad \pi(x) > 0 \quad \textit{and} \quad \sum_{x} \pi(x) = 1$$

$$\Pr(X^t = x \mid X^0 = x^0) \to \pi(x), \quad \forall x, x^0 \quad \text{as } t \to \infty.$$

Proof. See Feller (1968). □

The stationary distribution π is also called the *equilibrium* or *ergodic* distribution for the transition matrix Q. It follows that to sample from the stationary distribution $\pi(x)$, we can run a Markov Chain with transition matrix Q satisfying Equation (1.40) until the chain appears to have settled down to stationarity.

1.10 Markov Chain Monte Carlo Simulation

1.10.1 Constructing a MCMC Sampler

People have had success with MCMC on regular spatial structures. In image analysis, see §1.13. this spatial structure is represented by the pixels in an image which are spatially

next to each other being treated as neighbours in the conditional independence structure.

In this thesis we shall be investigating pedigrees, §1.14, where the spatial structure is represented on a graph and close relatives are used for conditional independence.

In pedigree analysis we wish to sample from a marginal posterior distribution of the genotypes. Ideally, we require a sequence of independent realisations but this is impossible. The best that we can do is to use a Markov Chain which depends only on the last step. The Markov Chain generates a sequence of dependent realisations, which from the *Ergodic theorem*, converges to a stationary distribution set up to be equal to the marginal distribution of interest. The chain is run from a random initial configuration but a large number of steps may be required before we obtain a sample from the stochastic process in order to ensure that x_t has a distribution close to the correct stationary distribution $\pi(x)$. We must also ensure that the chain forgets its initial configuration. Realisations from the Markov Chain are not independent because of the chosen initial structure and the correlation between successive realisations.

There are two solutions to these problems. Geyer (1993) advocates using a 5% burn-in which entails disregarding the first 5% of the number of samples that you wish to take before collecting your sample. The hope is that the Markov Chain will be closer to its stationary distribution by the end of this burn-in period. Geyer (1993) also suggests that using a single long-run rather than several short ones provides better estimates. The other approach is to sample every k^{th} realisation and provided k is large enough, we obtain samples that are virtually independent.

The second important objective of Markov Chain simulation is to evaluate accurate approximations of summaries of the stochastic process. These are of the form $E[\pi(X)]$ which can be written as

$$I = E[\pi(X)] = \int_{-\infty}^{\infty} \pi(x) f(x) dx. \tag{1.41}$$

To obtain estimates, we can use the average

$$\hat{I} = \frac{1}{n} \sum_{1}^{n} \pi(X_i) \tag{1.42}$$

where $X_1, \ldots, X_n$ are *iid* realisations from $f(x)$, since

$$\hat{I} \xrightarrow{P} I \tag{1.43}$$

by the Weak Law of Large Numbers.

In MCMC simulation, the *iid* realisations, $X_1, \ldots, X_n$ from $f(x)$ in Equation (1.42) are replaced by realisations from a Markov Chain with an identical stationary distribution. The *Ergodic* theorem ensures that Equation (1.43) still holds.

1.10.2 Advantages of MCMC

Efficiency and Use

Monte Carlo methods are usually the only method for numerical problems in a large number of dimensions. Methods for estimating an expectation say, $E[\pi(X)]$, include importance sampling, simulation techniques and factorising the distribution into lower dimensions from which sampling may be easier. The MCMC simulation method is slower than the other methods but in many cases it is the only possible alternative as the problem does not factorise and the importance sampling technique can be difficult to implement in higher dimensions. With increasing computer power, Monte Carlo methods have been applied to more complex systems.

Flexibility

The main advantage of simulation over analytical or asymptotic methods is its flexibility for non-standard cases and its ability to obtain almost exact solutions for arbitrary functionals of the process. It can be used to make interval estimates, deal with non-standard priors and likelihoods that would arise with missing data, simulate systems that can only be observed indirectly and assess simulation error. MCMC has been successfully used on well-known problems that have previously been intractable.

Credibility Regions

One of the main appeals of MCMC is the use of large samples from the chain to form credible regions for the posterior estimates directly from the empirical distributions. We can also use models that are believed to be most appropriate for the data even if they require non-standard likelihoods and non-conjugate priors.

Sensitivity Analysis

Sensitivity analysis is an important part of responsible statistical inference. Prior distributions can have a major influence on the posterior, so the sensitivity of the posterior to the prior is an important issue which MCMC methods can investigate.

1.10.3 Problems with MCMC

Convergence issues are important when dealing with computationally demanding simulation procedures such as MCMC. The large dimensionality of pedigree data sets or images makes monitoring the success of convergence difficult. The speed of convergence seems to decrease as the dimensionality increases (Green 1995) and this might be a problem as we consider traits with increasing numbers of alleles.

A canonical example of a mixture of two bivariate normal distributions displaced diagonally from each other (Tavener 1992) illustrates the problems that can arise with single variable updating in MCMC. As the distance between them increases, swaps between one

mode and the other become increasingly rare. For a finite number of realisations of the Markov Chain there may be few or no swaps but the chain is still irreducible. This is why MCMC may sometimes work badly and indicates that multi-modality must be checked for. Methods for solving this problem are given in §5.1.1 but the simplest is to use multiple starting points around the likelihood surface and see if convergence is to the same point each time. Increasing the number of starting points gives us increasing confidence in estimates based on any single starting point.

1.11 Possible MCMC Algorithms

There are several algorithms that can be used to generate a Markov Chain with a stationary distribution $\pi(x)$ for $x \in \Omega$. We consider the Metropolis algorithm, the Hastings algorithm and the Gibbs Sampler.

1.11.1 Metropolis Algorithm

The algorithm of Metropolis et al. (1953) was developed for use in physical chemistry in order to calculate interactions between two molecules and became widely used in statistical mechanics, see for example Hammersley and Handscomb (1964). The algorithm provides a way to reconstruct a Markov Chain $\mathbf{M}$ with a limiting distribution π. It is a single site update procedure and makes P a specified symmetric proposal matrix so that $p(x \rightarrow x') = p(x' \rightarrow x)$. At a given step, randomly draw a new state x' from the i^{th} row of P and with known probability $\alpha(x \rightarrow x')$ move from state x to state x'. Metropolis et al. (1953) set this acceptance probability to

$$\alpha(x \rightarrow x') = \min\left(1, \frac{\pi(x')}{\pi(x)}\right). \tag{1.44}$$

Algorithm 1.2. METROPOLIS: *Let π be a density and $P = \{p(x \rightarrow x')\}$ be a symmetric proposal matrix, so $p(x \rightarrow x') = p(x' \rightarrow x)$.*

- *If the chain is currently at $X^i = x$ then at each update, select a proposal x' randomly from the i^{th} row of p for the next update X^{i+1}.*
- *Accept the update (i.e. $X^{i+1} = x'$) with probability*

$$\alpha(x \rightarrow x') = \begin{cases} 1, & \text{if } \pi(x')/\pi(x) > 1, \\ \pi(x')/\pi(x) & \text{if } \pi(x')/\pi(x) \leq 1, \end{cases}$$

otherwise let $X^{i+1} = x$.

□

This is a Markov Chain with transition matrix $Q(x \to x') = \alpha(x \to x')P(x \to x')$ and is reversible since

$$\begin{aligned}
\pi(x)Q(x \to x') &= \pi(x)\min\{1, \pi(x')/\pi(x)\}P(x \to x') \\
&= \min\{\pi(x), \pi(x')\}P(x \to x') \\
&= \min\{\pi(x'), \pi(x)\}P(x' \to x) \\
&= \pi(x')\min\{1, \pi(x)/\pi(x')\}P(x' \to x) \\
&= \pi(x')Q(x' \to x). \qquad \square
\end{aligned}$$

This is a sufficient condition for the algorithm to yield a sequence of dependent realisations forming a Markov Chain with π as its stationary distribution. Notice that π need only be defined only up to a normalising constant since the constant cancels in $\pi(x')/\pi(x)$. The stationary distribution is unique if Q is irreducible and a sufficient condition for convergence is being able to move from any state to any other (Ripley 1987).

1.11.2 Hastings Algorithm

Hastings (1970) generalises Metropolis et al. (1953) to allow $P(x \to x')$ to be an arbitrary transition probability function; so if P is symmetric then the algorithm reduces to the Metropolis algorithm. Once again, $Q(x \to x') = \alpha(x \to x')P(x \to x')$.

Algorithm 1.3. HASTINGS: *Let π be a density and $P = \{p(x \to x')\}$ be a proposal matrix.*

- *If the chain is currently at $X^i = x$ then at each update, select a proposal x' randomly from the i^{th} row of p for the next update X^{i+1}.*
- *Accept the update (i.e. $X^{i+1} = x'$) with probability*

$$\alpha(x \to x') = \begin{cases} \min\{1, \pi(x')P(x' \to x)/[\pi(x)P(x \to x')]\} & \text{if } \pi(x)P(x \to x') > 0, \\ 1, & \text{if } \pi(x)P(x \to x') = 0, \end{cases}$$

otherwise let $X^{i+1} = x$. Note that $P(x \to x')$ is an arbitrary (i.e., not symmetric) proposal matrix.

$\square$

In order to show that this algorithm does generate a Markov Chain with the correct stationary distribution – that is for the *Ergodic* theorem to hold – it is sufficient to show that Q satisfies the *detailed balance* equation (1.39).

Proof of convergence. The proof of convergence is similar to the proof given for the Metropolis Algorithm. $\square$

The matrix Q can be anything so it is advisable to choose something that is a balance between convenience and performance. However α depends on π so we need to be able to calculate $\pi(x')/\pi(x)$ for any $x, x' \in \Omega$ with $Q(x \to x') > 0$. We set up Q so that the only transitions from x to x' are those for which $\pi(x')/\pi(x)$ is easy to calculate. Successive steps preserves the detailed balance, aperiodicity is never a problem as $q(x \to x) > 0$ for some x, but as we will see irreducibility needs to be checked.

1.11.3 The Gibbs Sampler

The Gibbs Sampler, which was formulated in Geman and Geman (1984), gives a time dependent version of the Hastings method. It consists of sampling sequentially from the conditional distributions of each parameter given all the others. For a discrete distribution over a small set of labels, the normalisation to a proper distribution is simple and so MCMC simulation is easy using the Gibbs Sampler.

Definition 1.28. A *sweep* of the configuration consists of working systematically once through each of the n elements in the configuration.

If we let $X = (X_1, \ldots, X_n) \sim \pi_X(x)$, we can use the following algorithm to generate realisations from $\pi_X(x)$.

Algorithm 1.4. GIBBS SAMPLER:

1. *Let* $t = 0$. *Start with any point in the multivariate distribution* $x^{(t)} = x^{(0)} = (x_1^{(0)}, x_2^{(0)}, \ldots, x_n^{(0)})$.

2. *Sweep through the* n *elements of* $x^{(t)}$ *to generate* $x^{(t+1)}$, *by sampling from the* full conditional *distributions.*

   - *Sample* $x_1^{(t+1)}$ *from* $\pi_{X_1|X_2,\ldots,X_n}(X_1 = x_1 \mid X_2 = x_2^{(t)}, \ldots, X_n = x_n^{(t)})$
   - *Sample* $x_2^{(t+1)}$ *from* $\pi_{X_2|X_1,\ldots,X_n}(X_2 = x_2 \mid X_1 = x_1^{(t+1)}, X_3 = x_3^{(t)}, \ldots)$

     $\vdots$ $\qquad\qquad$ $\vdots$
   - *Sample* $x_n^{(t+1)}$ *from* $\pi_{X_n|X_1,\ldots,X_{n-1}}(X_n = x_n \mid X_1 = x_1^{(t+1)}, \ldots, X_{n-1} = x_{n-1}^{(t+1)})$

3. *Increment* t *and go to Step 2.*

□

The vectors $x^{(0)}, x^{(1)}, \ldots, x^{(t)}, \ldots$ are a realisation from a Markov Chain. The transition probability Q from x to x' is

$$Q(x \to x') = \pi_{X_1|X_2,\ldots,X_n}(X_1 = x_1' \mid X_2 = x_2, \ldots, X_n = x_n, Y = y) \ldots\ldots$$
$$\ldots\ldots \pi_{X_n|X_1,\ldots,X_{n-1}}(X_n = x_n' \mid X_1 = x_1', \ldots, X_{n-1} = x_{n-1}').$$

This algorithm does converge to the required π and a proof is given in Geman and Geman (1984). We usually talk about visiting all the elements in a configuration rather

than looping over subscripts and for this we use the phrase 'sweeps of the configuration'. Thus, the Gibbs Sampler produces a whole new configuration after each sweep. It does this by updating the current value of the reconstruction for each individual with one sampled from the conditional distribution of interest for that individual and this move is always accepted.

1.11.4 Comparison of the Three Algorithms

The Gibbs Sampler and the Metropolis algorithm can be considered special cases of the general Hastings algorithm. As the Hastings algorithm satisfies detailed balance then all three do. For all three algorithms, a neighbouring state is generated and a decision is made to accept the move, or not, from the current state to the newly generated state. Gibbs Sampling always accepts the proposal whereas the Metropolis and Hastings algorithms have an acceptance probability.

Choosing between the MCMC algorithms depends upon the application. Gibbs is useful if sampling from the full conditional distributions is computationally cheap. Metropolis methods update single individuals by choosing between the current state and an alternative proposal and work well with either discrete or continuous variables. These methods are relatively easy to code whereas the Hastings algorithm is a little more difficult as the transition matrix is no longer symmetric. Both Metropolis–Hastings methods have had practical success but for efficient computing also depend on the acceptance rate.

1.11.5 Other Markov Models

In pedigree analysis, other approaches have been used for simulating the distribution of genotypes. Lange and Matthysse (1989) use the Metropolis algorithm to construct a random walk with stationary distribution matching the trait genotype conditional probabilities for a diallelic trait. Four transition rules define the neighbouring states, each of which refers to a *pivotal* individual in the pedigree in whose immediate locality the changes take place. These transition rules involve swapping the alleles of an individual and changing the sources of inheritance for either himself or his children. The Metropolis algorithm is used to generate the transition probabilities for the random walk. The authors prove the irreducibility of the Markov Chain defined for the special cases of a partially penetrant trait and a recessive trait.

Ott (1989) also develops a method for simulating genotypes at a marker locus given data at a disease locus.

1.12 Estimators for the Model

Our aim is to sample from a marginal posterior distribution in order to arrive at a point estimate for the configuration. There are several commonly used point estimators.

1.12.1 Marginal Posterior Mode Estimate

Definition 1.29. A Marginal Posterior Mode Estimate (MPM) is the point estimate of the configuration that maximises the posterior probability $\pi_{X_i|X_{obs}}(x_i \mid x_{obs})$ over x_i for each x_i in x_{unobs}.

This method, suggested by Marroquin, Mitter, and Poggio (1987), is to consider a loss function where each misclassified element incurs a penalty of l and the corresponding estimate, *Marginal Posterior Mode Estimate* (MPM), for element i is the x_i which maximises the posterior probability $P(X_i = x_i \mid x_{obs})$. To calculate the MPM estimate we obtain the marginal distributions from the posterior by running one of the Monte Carlo Markov Chain simulation methods above and note the frequency with which each of the n values occurs for each element. After the chain has finished, the estimate for each element, i, is taken to be the most frequently occurring value of x_i. Clearly, the approximation will improve with an increasing number of simulations. In image analysis, the colour of a pixel is represented by a small discrete sample space and so this method works very well. A method similar to this is used in the `Pedpack` package of programs (Thomas 1991) for pedigree analysis which is used in calculations in the remainder of this thesis. Instead of taking the most frequently occurring value as our estimate, we are interested in approximating probabilities or frequencies which are continuous and so can gene count over the realisations to obtain our estimates. As we are gene counting, this method has the added advantage of smoothing out the effects of any unusual simulations thrown up by any of the MCMC algorithms. These methods are easy to implement and can be thought of as sampling techniques.

Other techniques use combinatorial optimisation. To help drive the MCMC algorithms to convergence in a finite amount of time we nest them within an optimisation algorithm. We consider the stochastic algorithm simulated annealing (SA) below.

1.12.2 Maximum A Posteriori

Definition 1.30. A Maximum A Posterior (MAP) estimate is defined to be the point estimate of the configuration $x \in \Omega$ that maximises the posterior distribution $\pi(x)$

It is computationally infeasible to try this maximisation by an exhaustive search due to the huge number of possible configurations, m^n, where m is the number of elements and n is the number of possible values for each element, and so, Kirkpatrick et al. (1983) developed the stochastic relaxation algorithm or *simulated annealing* algorithm, which was suggested in numerical calculations by Metropolis et al. (1953).

1.12.3 Simulated Annealing

Kirkpatrick et al. (1983) discovered an analogy between minimising the cost function of a combinatorial optimisation problem and the slow cooling of a solid. They called this algorithm *simulated annealing*. The process attempts to mimic a technique in physics

where if a molten metal is allowed to cool sufficiently slowly then the atoms line themselves up and form a pure structure avoiding imperfections in the solid. The crystal achieves the minimum energy state for the system.

Combinatorial optimisation involves attaching a real number to each of a finite or countably infinite number of states using a cost or *energy function*. The objective is to search the state space to find the state with the minimum energy. The modification that enables simulated annealing to optimise rather than sample is the raising of the target distribution to higher and higher powers over the course of the algorithm which places an increasing probability at the globally optimum state. By using the functional of interest in the Metropolis algorithm as the energy function in the simulated annealing algorithm and slowly increasing the power, Kirkpatrick et al. (1983) found a general solution to combinatorial optimisation problems. The power is increased by reducing a parameter, T, called the temperature parameter. This odd notation attempts to ensure similarity between the physical and mathematical aspects of the algorithm. A starting configuration is needed to initialise the algorithm.

Algorithm 1.5. SIMULATED ANNEALING:

- *Initialise to any convenient configuration.*
- *Simulate from a process with the distribution*

$$f^T\{\pi(x)\} = \left[f\{\pi(x)\}\right]^{1/T} \Big/ \sum_{x' \in \Omega} \left[f\{\pi(x')\}\right]^{1/T}. \tag{1.45}$$

- *Complete a* sweep *of the configuration by sequentially updating each of the elements in the configuration.*
- *After each sweep of the configuration the parameter T is reduced, such that,*

$$T_k \geq T_{k+1} \qquad \textit{and} \qquad \lim_{k \to \infty} T_k = 0,$$

according to a predefined schedule. □

We define a *temperature schedule*, $\{T_k\}$, for $k = 1, 2, \ldots$, where $\{T_k\} \to 0$. At step k a sample using the family of MCMC algorithms is taken from the distribution at temperature T_k. As $t \to 0$, our estimate of the true pedigree or image concentrates on the mode of the posterior probability. In theory, if the temperature is decreased at a sufficiently slow rate, we arrive at the MAP estimate. At $T_k = 0$, the distribution will have positive mass only at the maximising x and so if we draw a sample from this distribution, we are selecting from the x we wish to make inferences about.

Remarks.

- The crux of the algorithm is that it might accept a move to a state with a higher energy function as a means of escaping from a local minimum.

- If the starting temperature is sufficiently high, the starting point is not critical because the algorithm can escape from local minima.
- The algorithm relies on a process of iterative improvement. This makes it easy to implement but it is slower that some *greedy* algorithms, which can only converge to local minimums.
- The algorithm has the potential to reach, but not to detect, the global maximum of the posterior distribution.
- Improved estimates of the maximum can be obtained if you record the largest estimate produced by the algorithm and not just the final estimate.

Temperature Schedules for Simulated Annealing

The family of Metropolis–Hastings algorithms requires many iterations to attain convergence to the stationary distribution but this is simple to implement on a computer; however, the choice of T_k and the cooling rate is all important. Geman and Geman (1984) and Hajek (1988) suggest that the cooling of T_k should be very slow in order to achieve convergence. In theory, the temperature parameter should be decreased on a log schedule but in practice we can only afford to decrease the temperature towards zero in finite jumps. Consequently, we can only expect to find a local minimum. However, if the starting temperature is high enough, we expect to explore the state space sufficiently to find a good local minimum of the functional of interest. Stander and Silverman (1994) consider four families of temperature schedules: straight, geometric, reciprocal and logarithmical. Examples of these are shown in Figure 1-14.

The simulated annealing approach has the advantage that we can explicitly control the number of sweeps, which is the parameter to which the CPU time is most sensitive. In §2.2 and §2.3 we use a geometric cooling schedule to order the summing out of individuals in the peeling algorithm to reduce the cutset and hence memory requirements because it is found to work better in practice than other temperature schedules, not cooling too quickly or too slowly.

1.13 Image Analysis

Markov Chain simulation techniques began to be used in image analysis, where the observed image or *record*, Y, depends on an underlying true image X, so

$$\begin{aligned} \text{Record} &= f(\text{Image}) + \text{Noise} \\ Y &= f(X) + \varepsilon \end{aligned} \tag{1.46}$$

and became applicable by examining the problem on the small scale where each small region, *pixel*, is likely to be similar in colour to the regions near it. This can then be

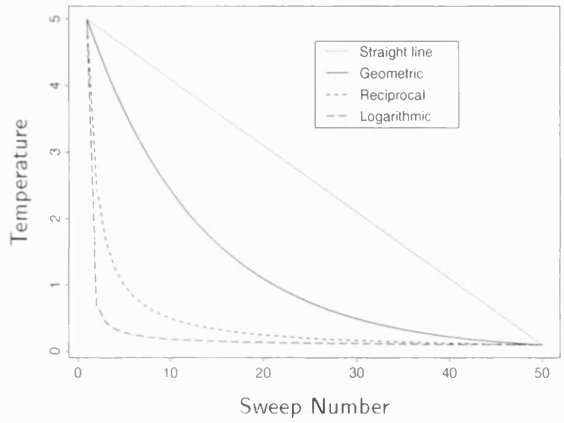


Figure 1-14: FAMILIES OF TEMPERATURE SCHEDULES: Some examples from the families of temperature schedules given in Stander and Silverman (1994) are shown here, where there are 50 sweeps with a starting temperature of 5. The logarithmic schedule is theoretically superior but, for a finite number of sweeps, the temperature has been found to drop too rapidly to fully explore the state space, as does the reciprocal schedule. On the other hand, the straight line was felt to decay too slowly. Thus the geometric schedule is preferred.

formulated into a statistical model which is briefly developed below to motivate intuition in the pedigree problem.

The MRF is defined by the local dependence of an individual pixel on the pixels in a neighbourhood around it as shown in Figure 1-15.

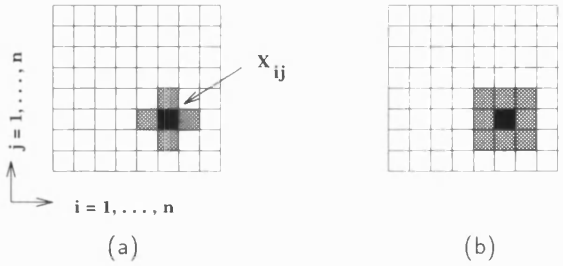


Figure 1-15: NEIGHBOURHOOD SCHEMES IN IMAGE ANALYSIS: The representation of a picture in a pixel lattice. (a) Shows a first-order neighbourhood scheme for pixel x_{ij}. (b) Shows a second-order neighbourhood scheme for pixel x_{ij}.

If we consider an image, it is represented by a finite two dimensional grid containing m pixels. The true scene is denoted by $X \stackrel{\text{def}}{=} (X_1, X_2, \ldots, X_m)$ but the observed data or *record*, denoted $Y = (y_1, y_2, \ldots, y_m)$ may be degraded due to blurring or noise.

Geman and Geman (1984) use a Bayesian approach. They formulate the posterior distribution of the scene X given the data or record Y; using Bayes' Theorem, this distribution can be expressed in terms of the likelihood of Y given X and a prior distribution for X, as in Equation (1.35).

Using one of the MCMC algorithms, say the Gibbs Sampler, to recreate a blurred or noisy image is now straightforward. The process starts at the top left corner of the image. For that pixel, the colours of its neighbouring pixels are looked at and a decision is made what colour to make the pixel by drawing from its conditional distribution given

its neighbours and the record. The process is then repeated for all the pixels in the image after which we have completed one sweep of the configuration. To get a point estimate of the image we do this many times and either keep counts on the colour of each individual pixel with our final estimate for each pixel being the colour that occurred the most often, or nest the process within an optimisation algorithm, such as simulated annealing, which gradually makes changing the colour of an individual pixel less and less likely and so the picture gradually 'freezes'.

1.14 Pedigree Analysis

Using the MRF idea, MCMC algorithms can be implemented not only on a lattice but to any network with well-defined relationships which are not spatial. Sheehan (1990) applies the method to pedigree analysis.

1.14.1 Neighbourhood

The graph of the pedigree, $\mathcal{G}$, is the region of interest, the individuals represent the pixels, the genotypes replace the true image and the phenotypes are the records which provide information on the underlying genotypes. Sheehan (1990) shows that if the neighbourhood, δ_i, of an individual i, consists of parents, spouses and offspring, as in Figure 1-16, then

$$P(X_i = x \mid \mathbf{x}_{\mathcal{G}\setminus i}) = P(X_i = x \mid \mathbf{x}_{\delta_i}) \tag{1.47}$$

and so the definition of a MRF is satisfied. However, the defined field does not satisfy

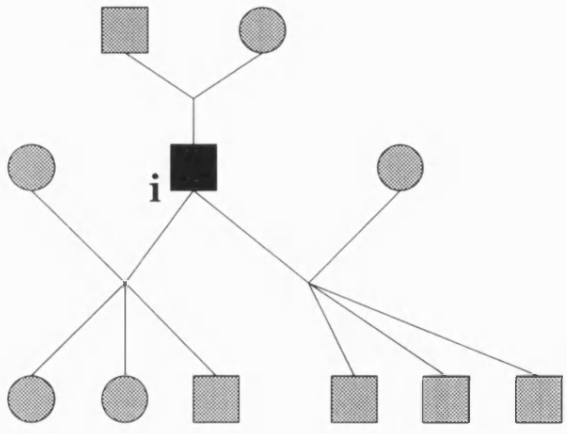


Figure 1-16: NEIGHBOURHOOD FOR PEDIGREES: For the neighbourhood system for individual i in a pedigree it is convenient and obvious to use close relations as these are the individuals that will affect i's genotype. So we use parents, spouses and offspring of individual i.

the positivity condition for Equation (1.32) since certain parental genotypes combinations exclude the possibility of certain genotype combinations in the offspring.

Hammersley and Clifford had tried to prove that the positivity constraint was irrelevant but the reason for their failure was shown in Moussouris (1974), who found a counter-example. He proceeded to discuss what happens in the absence of the condition and established criteria for a system in which the positivity does not hold.

Sheehan (1990) stresses that these problems only occur when one is trying to formulate a MRF; the statistical geneticist has already been provided with a fully specified locally dependent field.

1.14.2 Irreducibility

Irreducibility is a more difficult problem to overcome in pedigrees than in images. Convergence in distribution of the Gibbs Sampler to the correct posterior distribution of genotypes given the phenotypes depends on the irreducibility of the relevant Markov Chain. This is clearly true in the case where the positivity holds but as we have stated above, not all configurations are legal for pedigrees. All that is important is that all *feasible* configurations communicate because as long as we start with a feasible configuration, the Gibbs Sampler will converge to the required posterior distribution.

If we consider a two allele case, by working through the pedigree it is possible to assign every individual a heterozygous genotype which, because of the laws of Mendelian inheritance, is feasible and guarantees all configurations can be visited from any other in a finite number of sweeps by updating one individual at a time. Of course, a totally heterozygous configuration may not be possible as the configurations also have to be consistent with any observed data. Sheehan (1990) gives the following example.

Example 1.9. FAILING IRREDUCIBILITY:

Suppose there is a disease in the population where the homozygotes are phenotypically indistinguishable as *unaffected* and the heterozygotes are *affected*; and so has the penetrance matrix given in Table 1.1.

	UNAFFECTED	AFFECTED
AA	1	0
BB	1	0
AB	0	1

Table 1.1: PENETRANCE MATRIX SHOWING REDUCIBILITY: This table shows the penetrance matrix for a simple disease model which is not irreducible.

Further suppose that a tested individual is found to be affected but his parents are known to be normal; then as shown in Figure 1-17, two configurations are possible due to

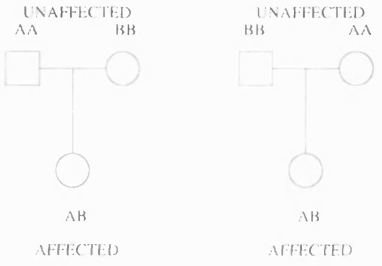


Figure 1-17: FAILING IRREDUCIBILITY CONFIGURATIONS: Two feasible genotype configurations for the diallelic genetic model given in Table 1.1.

the way that we can assign the genotypes from their homozygous parents. Now, each of

these feasible configurations cannot be reached from the other by updating one individual at a time because to maintain the laws of Mendelian inheritance the male parent say must be homozygous and so could only change to BB but this then breaks the rules of inheritance for the offspring. □

Clearly, irreducibility of the Markov Chain of interest requires that we are able to pass through this state and this led Sheehan to prove the following theorem.

Theorem. IRREDUCIBILITY: *For any diallelic trait the Markov chain is irreducible provided that for any phenotype ϕ, if $\rho(\phi \mid AA) > 0$ and $\rho(\phi \mid BB) > 0$, then $\rho(\phi \mid AB) > 0$.*

Thomas (1995) used the Gibbs Sampler method to estimate missing data and demonstrates the method on a partially incomplete set of genotypic information for nine diallelic traits of the Przewalski's horse pedigree, but other higher allele traits were not considered due to the problems of finding an initial configuration and irreducibility. Sheehan (1991) investigated a diallelic trait on the complex, highly-looped pedigree of the Polar Eskimos.

1.14.3 Relaxation Sampling

A solution to problems connected with irreducibility and obtaining the initial configuration was given in Sheehan (1990) and implemented in Sheehan and Thomas (1993). The authors relax the transmission probabilities and give every phenotype with zero transmission a small positive probability of being realised by every genotype. So

$$\tau_*(i_k, i_{f_k}, i_{m_k}) = \begin{cases} \tau(i_k, i_{f_k}, i_{m_k}) & \text{for } \tau > 0, \\ \gamma > 0 & \text{for } \tau = 0. \end{cases} \tag{1.48}$$

The resulting model is irreducible, as all states are obtainable from every other by single updates, but configurations are produced that contravene the laws of Mendelian inheritance; these are rejected. The authors show that the legal simulations produced from this incorrect model have a stationary distribution that is equal to the correct model. When using this rejection method the rejection rate has to be carefully considered. The smaller the γ, the more correlated the feasible configurations as the rejection rate is low. The larger the γ, the larger the rejection rate which decreases correlation but increases the simulation time. Generally the more genotypes in the genetic system, the larger the rejection rate for a particular γ.

The method appears to be some kind of rejection sampling but is detailed in Hastings (1970) as importance sampling. Let $\Omega = \{\text{all legal configurations of alleles on pedigree}\}$. If we define $\pi^*(x)$ to be positive on $\Omega^* = \{\text{all configurations of alleles on pedigree}\}$, which corresponds to the model obtained relaxing the transmission probabilities. It is possible to MCMC sample $\pi^*(x)$ and then use these samples reweighted. For example

$$E_\pi(f) = \sum_{\Omega} f(x)\pi(x) \tag{1.49}$$

and because all the contributions of π in $\Omega^\star$ are zero

$$E_\pi(f) = \sum_{\Omega^\star} f(x)\pi(x) \tag{1.50}$$

$$= \sum_{\Omega^\star} f(x)\frac{\pi(x)}{\pi^\star(x)}\,\pi^\star(x) \tag{1.51}$$

$$= E_{\pi^\star}(f(x)\pi(x)/\pi^\star(x)). \tag{1.52}$$

From Hastings (1970), if we set up the Markov Chain using the distribution $\pi^\star$ instead of π we can estimate $I = E_\pi(f)$ by

$$\hat{I} = \frac{\sum_{t=1}^{N}[f(X^t)\pi(X^t)/\pi^\star(X^t)]/N}{\sum_{t=1}^{N}[\pi(X^t)/\pi^\star(X^t)]/N}. \tag{1.53}$$

So we can now run a Gibbs Sampler on the genotypes of individuals in a pedigree by the method outlined in §1.13. As we have set up an irreducible Markov Chain, we can start from any configuration and by running a Gibbs Sampler on the genotypes of the pedigree by looking at each individual in turn and choose his new genotype from his conditional distribution for $\Omega^\star$ given his neighbours' genotypes. We use all of the X^ts but weight $\pi(X^t)/\pi^\star(X^t)$ is zero for $X^t \notin \Omega$. Note that for all $x \in \Omega, \pi(X)/\pi^\star(X) = c$ for some constant c. So from Equation (1.53)

$$\hat{I} = \frac{\sum_{t=1}^{N}[f(X^t)\ I(X^t \in \Omega)\ c]/N}{\sum_{t=1}^{N}[I(X^t \in \Omega)\ c]/N} \tag{1.54}$$

$$\hat{I} = \frac{\sum_{t=1}^{N} f(X^t)\ I(X^t \in \Omega)}{\sum_{t=1}^{N} I(X^t \in \Omega)}. \tag{1.55}$$

1.15 Density Estimation

During the course of this thesis we use a Bayesian framework as a computational tool for non-Bayesian likelihood inference. The realisations produced give points on the sample space and so need density estimation to get some estimates of the MLEs and the likelihood surfaces. A brief review of density estimation is given in this section. For a fuller account see Silverman (1986) or Scott (1992).

1.15.1 Histograms

The oldest, most widely used, and easiest form of density estimation is the histogram. After choosing an origin x_0 and a *bin-width* h, we define the *bins* of the histogram to be the intervals $[x_0 + mh, x_0 + (m+1)h)$ for positive and negative integers, m. The histogram

is then made up from the number of observations in the same bin as x, *i.e.*,

$$\hat{f}(x) = \frac{1}{nh}(\text{number of } x_i \text{ in the same bin as } x). \tag{1.56}$$

To construct this density estimator we have to choose the origin and a bin-width. Although the origin can influence the density estimate, it is the choice of bin-width which mainly affects the amount of smoothing in the process.

Example 1.10. DIFFERENT BIN WIDTHS: If we simulate 100 observations from a standard normal distribution we can get the results shown in Figure 1-18 for different bin-widths.

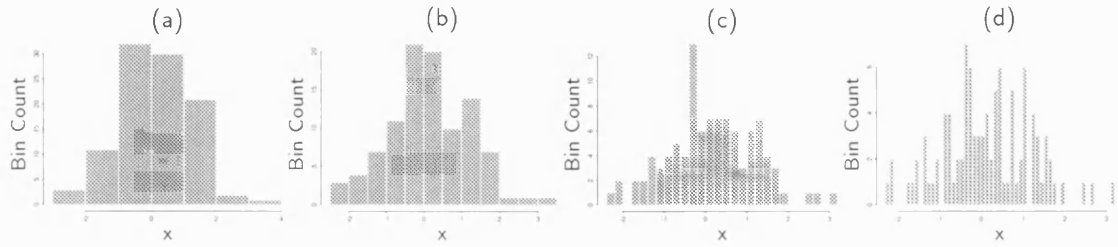


Figure 1-18: HISTOGRAMS: The density estimates for a set of 100 simulations from a standard Gaussian distribution is shown for different bin-widths.

□

A more general approach is allowing the bin widths to vary.

$$\hat{f}(x) = \frac{(\text{number of } x_i \text{ in the same bin as } x)}{n(\text{width of bin containing } x)}. \tag{1.57}$$

The choice of bin widths can be done *a priori* or based on the data.

The advantages of the histogram as a density estimator are that it is well known and easy to use. Some of the disadvantages are that the estimate it produces is discrete, it varies with the choice of origin and it presents multi-dimensional data with difficulty.

1.15.2 The Naive Estimate

If f is the *probability density function* of X then

$$\Pr(X = x) = f(x) = \lim_{h \to 0} \frac{1}{2h} \Pr(x - h < X < x + h). \tag{1.58}$$

So a natural estimator of f is to choose

$$\hat{f}(x) = \frac{1}{2hn}[\text{number of } x_1, x_2, \ldots, x_n \text{ falling in } (x - h, x + h)]. \tag{1.59}$$

This is the *naive estimator*. A different definition is

$$\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} w\left(\frac{x - x_i}{h}\right), \text{where } w(x) = \begin{cases} 1/2 & \text{if } |x| \leq 1 \\ 0 & \text{otherwise.} \end{cases} \tag{1.60}$$

In effect, a box of width $2h$ and height $(2nh)^{-1}$ is placed at each data point and the estimate is formed by summing the boxes. So, the naive estimator can also be seen as a histogram where every point is the centre of one of the histogram bins. This removes the problem of choosing an origin but the naive estimator still requires a bin width. Unfortunately, the naive estimator is also discrete and so a more general approach is needed.

1.15.3 The Kernel Estimator

If the weight function, $w(x)$, is replaced by a function K where

$$\int_{-\infty}^{\infty} K(x)dx = 1 \tag{1.61}$$

then a generalised form of a naive estimator called a *kernel estimator* can be defined by

$$\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right) \tag{1.62}$$

where h is called the *bandwidth* or smoothing parameter and K is called the *kernel function.*

The Kernel Function

It can be seen from the definition that if the kernel function is continuous and differentiable then the density estimator, $\hat{f}$, is continuous and differentiable to the same degree. Furthermore, if the kernel function is a p.d.f. , then f is a p.d.f. .

The most interesting kernel functions have been found to be symmetric, unimodal probability density functions such as the normal density. The standardised *normal kernel* or *Gaussian kernel* function, denoted by K_N, is also a continuously differentiable function and is defined by

$$K_N(x) = \frac{\exp(-\frac{1}{2}x^2)}{\sqrt{2\pi}}, \qquad \text{where } -\infty < x < +\infty. \tag{1.63}$$

The Bandwidth

The bandwidth parameter influences the shape of the estimated p.d.f. . It is similar to the choice of bin-width or class-width in a histogram. Its visual effect is to govern the degree of smoothing in the estimate of the density. In particular, when $h = 0$ no smoothing occurs so the estimated density will be equal to the set of observations. As h increases,

the estimated density will become smoother. For larger values of h, the density will be very smooth but by then the fit may be poor. For K_N, most of the information used to calculate the density at any point comes from the range $\pm 3h$ as the bandwidth is the standard deviation of K_N.

1.15.4 Multivariate Data

So far we have restricted ourselves to univariate data but most of the important applications of density estimation involve the analysis of multivariate data. It is easy to understand a contour plot or perspective view of a two-dimensional density function but presentational difficulties make it unlikely that density estimates will be useful for exploratory purposes. However, if the intention is not to look at the density function but to use it as part of some broader statistical technique, then presentational aspects become less important and it may be useful and necessary to estimate densities in a higher dimensional space.

Histograms

The arguments for using density estimates rather than histograms become much stronger in two or more dimensions. The construction of a multivariate histogram requires the specification of not only the size of the bins and the origin of the system of bins, but also the orientation of the bins. Without experience presentational difficulties, such as the inherent 'block' effect, make it difficult to grasp the structure of the data. Because of their continuous nature, bivariate density estimates constructed using continuous kernels are much easier to understand when presented as a perspective plot.

A further difficulty with histograms is that if the bin-width is small enough to have any chance of catching local information, then, even in 2D, the total number of bins becomes so large that spurious effects are likely to dominate.

The Kernel Estimator for Multivariate Data

The kernel density estimator can be extended to higher dimensions. If $\{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_n}\}$ is the given multivariate set of observations whose underlying density is to be estimated, the multivariate kernel density estimator with kernel K, bandwidth h and the dimension d is defined by

$$\hat{f}(\mathbf{x}) = \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right), \qquad \text{where} \int_{\mathcal{R}^d} K(\mathbf{x}) d\mathbf{x} = 1. \tag{1.64}$$

Often K will be a simple symmetric unimodal p.d.f. such as the standard multivariate normal p.d.f. .

Bounded Domains

Frequently, the natural domain of the density that we wish to estimate is not over the whole of $\mathcal{R}^d$. Most often, for univariate data, measurements only have meaning if they are positive quantities and so are bounded by the coordinate axes. For multivariate data, the space could be restricted by some condition, *e.g.*, allele frequencies summing to 1. For exploratory purposes the boundary effect can be ignored but for other purposes estimates which give weight to a region of the space which is prohibited are unacceptable.

One way to ensure that $\hat{f}(x)$ is zero for these forbidden spaces is just to estimate in the bounded region of interest which we denote by $\mathcal{I}$. However, if we use an approach like the kernel method, which usually produces estimates that are probability densities, then the estimates will no longer sum to unity. Additionally, points near the boundary will have much less effect than those away from the boundary, resulting in an underestimation of density at the boundary.

It is possible to adapt methods designed to work on the whole space. Suppose we augment the data by adding the reflections of all the points in the boundary, to give the set $\{X_1, X_{-1}, X_2, X_{-2}, \ldots\}$. If a kernel estimate f^* is constructed from this data set of size $2n$, then an estimate based on the original data can be given by putting

$$\hat{f}(x) = \begin{cases} 2f^*(x) & \text{for } x \in \mathcal{I} \\ 0 & \text{for } x \notin \mathcal{I} \end{cases} \tag{1.65}$$

This corresponds to a general weight function estimator for x and $y \in \mathcal{I}$,

$$w(x,y) = \frac{1}{h}K\left(\frac{y-x}{h}\right) + \frac{1}{h}K\left(\frac{y+x}{h}\right). \tag{1.66}$$

Provided that the kernel is symmetric and differentiable, the estimate will always have zero derivative at the boundary. We do not have to reflect the whole data set since if X_i/h is sufficiently large then the reflected point will not contribute to the calculation of $f^*(x)$ for $x \in \mathcal{I}$, and so we only reflect points that are near the boundary. For example, if K is the normal kernel then there is no real justification in reflecting points X_i, for which $| X_i - X | > 4h$.

1.16 Objectives of the Thesis

1.16.1 Objectives

The objective of the thesis is to provide some information about the validity of two different management programs on the pedigree of the Przewalski horse. For this pedigree we have information on 16 traits ranging from two to six alleles. We summarise our objectives and methods as follows.

Ancestral Inference: The primary objective is to make accurate ancestral inference on the founders of the pedigree, as we need to investigate these individuals to help answer the management questions of interest. This involves finding the MLEs of the allele frequencies of the founders. The existing method for this is peeling and in Chapter 2 we introduce peeling as the E–step of an EM algorithm which finds the MLEs more efficiently than repeatedly running the peeling process over a fine grid of allele frequencies. A problem exists though in that accurate uncertainty bounds for these estimates are difficult to obtain using this method. Peeling is also a memory eating algorithm and this method for finding exact answers can only be used on the two and three allele traits; in fact, even for these, we have to run the optimisation algorithm, simulated annealing, to reduce the memory requirements. So, in Chapter 3 we replace the peeling algorithm (the E–step of EM) with a MCMC simulation method, the Gibbs Sampler, which samples from a stationary distribution equal to the marginal posterior distribution of interest; we refer to the result as the *Gibbs EM algorithm.* Unfortunately, unlike image analysis where irreducibility is guaranteed, irreducibility fails for multi-allele traits due to the basic rules of inheritance. We solve this by using a relaxation algorithm which samples from the wrong stationary distribution but one which is proportional to the correct stationary distribution. It produces illegal configurations but these are rejected. This technique allows us to obtain calculations on all the multi-allele traits. In this thesis we are mainly interested in the MLEs or marginal posterior modes. For the two and three allele traits, we have the luxury of being able to compare the exact results obtained from the peeling EM algorithm with the simulated results from the Gibbs EM algorithm to see how well the simulation is performing. We investigate the effect of the run parameters on the rejection rate and serial correlation of the new sampling algorithm, and concerns about its inability to sample from the whole of the subspace of interest.

Accurate uncertainty bounds are still difficult to obtain due to the nesting of the Gibbs Sampler within an EM algorithm structure. To try and improve on this method, in Chapter 4 we use the Bayesian framework. By using a constant *Dirichlet* prior distribution we use the framework as a computational likelihood engine. We use the Gibbs Sampler to generate samples from the posterior distribution which is proportional to the likelihood. Obtaining marginal means and uncertainty estimates is now straightforward; however, we are also interested in MLEs and different forms of density estimation are also examined.

Density Estimation: As mentioned in the above item, we are going to have to use density estimation in order to obtain estimates of the MLEs in high dimensional space. This again is computationally costly and existing algorithms for density estimation cannot handle too many points due to memory requirements. So, using the increased speed of looping in `S-Plus`, we have created a new way of calculating the Gaussian kernel estimate which can in theory handle an unlimited number of points.

Algorithmic Design: Due to the heavy computational costs that are involved in the simulation procedures it is vital that we design any algorithms to be efficient. We structure the MCMC algorithms to be strictly local in nature making the computational aspects of the algorithms easier to understand and visualise, and consequently to implement. It also means that the algorithms are efficient in their use of CPU time and storage requirements.

Computational Aspects: Throughout the thesis we have competing factors. For example, we would like to run the Markov Chain for as long as possible with as high a parameter in the relaxation algorithm as possible, and we would like to collect as many realisations as possible from the full Bayes method to input into the density estimation procedure, but both of these mean colossal run times and computational cost. So we have to balance the factors against computational cost and be prepared to put up with lower numbers of realisations or a greater serial correlation than we would like in order to get estimates in a reasonable time.

1.16.2 Topics not Covered

Mutation and Recombination: In this thesis we do not consider mutation or recombination of genes as they pass through the pedigree. Both of these effectively change the genome. Mutation is where a gene changes due to some action like environmental pressures. Recombination involves the mixing of the parents' genetical material and instead of getting one of your maternal chromosomes you inherit a chromosome which is made up of a mixture of both.

Linkage: Linkage analysis has to do with the places along the genome where recombination occurs. It is another area which is computationally demanding especially when dealing with multi-allele locus. The introduction of MCMC simulation methods has been successful, for example Ploughman and Boehnke (1989) investigate the power of a linkage study on the basis of partial information prior to embarking on huge marker studies, but there are still problems to overcome. Using the method of Thomas (1992), initial linkage analysis was carried out between all pairs of two allele traits and all pairs of three allele trait for the PH pedigree. There was a suggestion of linkage between a few trait pairs but this remains to be investigated.

Metropolis–Hastings Algorithms: For this thesis, the Gibbs Sampler is computationally the easiest of the three MCMC algorithms to implement. Sampling from the marginal distributions is simple and for efficient computation, as we already have a rejection parameter involved with the relaxation method, we do not feel the introduction of another rejection probability associated with the Metropolis–Hastings algorithms would be beneficial in speeding convergence. However, this could be investigated in possible future work.

Alternative MCMC Methods: There have been doubts as to whether the simple Gibbs

Sampler converges quickly enough to the stationary distribution to provide accurate estimates in reasonable computer time. Other methods that make the Markov Chain mix more rapidly and hence speed convergence have been implemented and are outlined in §5.1.1. However, they have not been implemented in this thesis as we have found that for this pedigree, the simple Gibbs Sampler provides good estimates for reasonable computer cost. Alternative MCMC sampling methods remain of interest for possible future work.

Chapter 2

A Peeling EM Algorithm

Nature is not only odder than we think,
but it is odder than we can think.
(J B S Haldane 1893 - 1964)

2.1 The Przewalski Horse Pedigree

The remainder of this thesis will use the pedigree of the Przewalski horse as a data set on which to base methods and calculations.

2.1.1 History

The Przewalski horse, *Equus Przewalski*, was first described by the Russian zoologist Poliakov (1881). Colonel Nikolai Przewalski obtained a skull and skin of a wild horse in the Mongolian district of Kobdo and sent specimens to Poliakov, who named the horse after the Polish-born Colonel in the Tsar's army. The Przewalski horse is the only true wild horse as populations of feral horses at numerous locations around the world are all derived from the domestic horse, *Equus caballus*.

Between 1899 and 1903, 53 young Przewalski horses, which survived the hardships of long journeys from Mongolia, were initially transported to several zoos and private parks in North America and Russia, but several zoos in Western Europe also managed to get hold of some of the horses. Blood from only eleven of these horses contributes to the present breeding stock and these were found in four breeding institutions: North America, Russia, Great Britain and Prague; however at Prague, a Przewalski stallion was cross-bred with founder DOM. a Mongolian domestic mare (Bouman 1982) and so introduced domestic genes into the population. Breeding continued independently at each of the establishments until the Second World War resulting in four highly inbred lines.

The war caused enormous damage to Przewalski horse. The records at the British breeding centre at Woburn were destroyed after the building was bombed and this together with the frequent movement of horses between Woburn and London Zoo meant that the

new breeding records were based on anecdotal evidence and suggestion. More importantly, only 31 horses of the breeding stock survived and out of these only nine horses, three stallions and six mares, were to produce offspring. However, new genes were introduced into the population in 1947 after a mare 231, the final founder, was captured in a different area of Mongolia from the early founders and was successfully bred from. Since then, concern about the high level of inbreeding has led to cooperation between zoos to increase the amount of interbreeding in order to reduce inbreeding and increase the population. These efforts have been successful, with the current captive population approaching 2000 individuals, (Volf 1991).

Since the late seventies, 147 of the horses have been tested for one or more of sixteen allozyme polymorphisms (Putt and Fisher 1979, Mace and Whitehouse, personal communication).

Figure 2-1 gives a marriage node graph of that part of the pedigree ancestral to the horses tested for at least one of the allozyme polymorphisms. The squares represent males, the circles females and the diamonds individuals whose sex was untested and who have no offspring. The diamonds may, in fact, represent several siblings. Matings are represented by black dots which are connected upwards to parents and downwards to offspring. The individuals tested for polymorphisms are indicated in grey.

2.1.2 Management Programs

The breed is now believed to be extinct in the wild since there have been no confirmed sightings for nearly 25 years (Ryder and Wedemeyer 1982) and so the entire population is held in captivity and relies entirely on man for its future existence. We have a great deal of information about the zoo pedigree due initially to Dr Erna Mohr and latterly to Dr Jiri Volf. These facts have meant that the pedigree and its management have taken on special importance. It acts as a blueprint for the development of methods for preserving the ever increasing number of other endangered species in captivity with the long term aim of re-introducing these animals back into the wild. However, the pedigree has many interesting features and problems which make the management of the population far from easy, and there are disagreements as to the best approach.

Gene Pool

The whole population being descended from only 13 founders means that we have to manage a small gene pool. There is evidence demonstrating that considerable genetic variation was present in the Mongolian populations of Przewalski's horse that provide 12 of these founder animals. Several authors, for example Mohr (1959, 1969), have discussed the external morphological variation in phenotypes, and anecdotal evidence, such as photographs of some of the founders and their descendants, is available and clearly show a large variation in coat colouration, eye colour, head shape and tail. More recently, studies and analysis of blood have detected considerable genetic variation within the genome of the wild horse (Putt and Fisher 1979, Ryder et al. 1979). As considerable inbreeding has

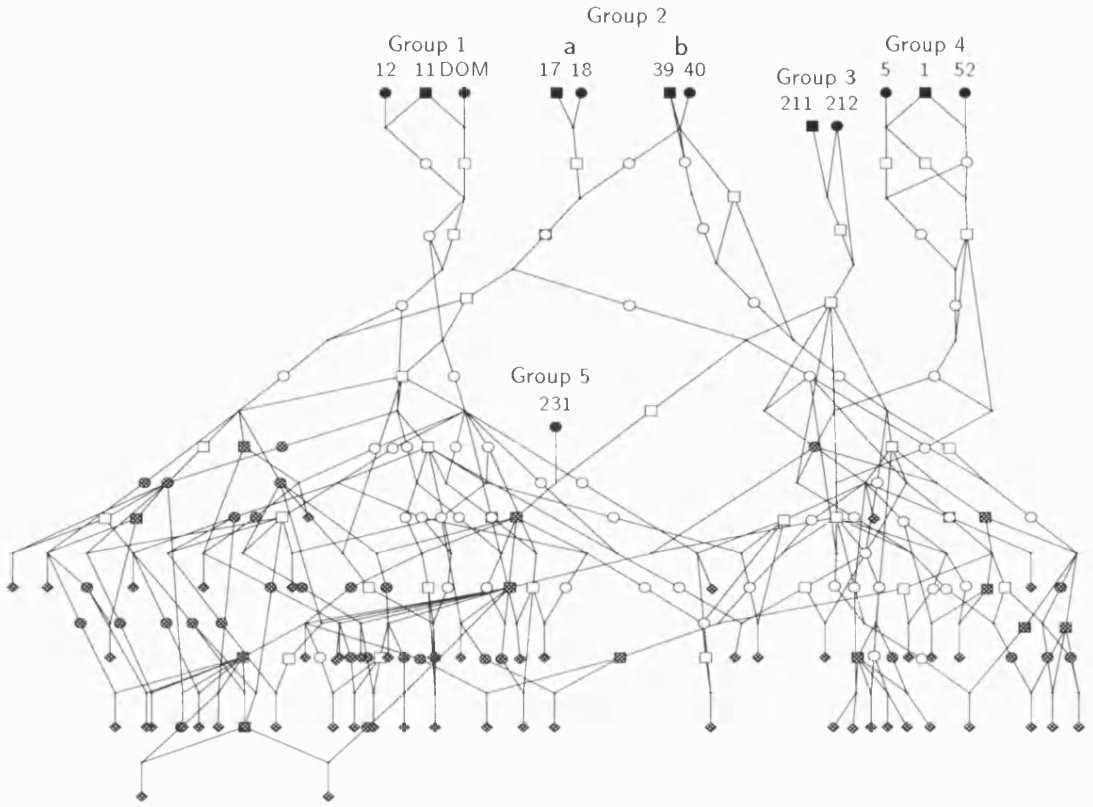


Figure 2-1: THE PRZEWALSKI HORSE PEDIGREE: The figure shows a marriage node graph of the part of the pedigree ancestral to the horses tested for at least one of the allozyme polymorphisms since the late seventies. In the pedigree squares represent males, the circles represent females and diamonds represent an individual or multiple individuals whose sex is unknown. The black shaded individuals are the founders who have been split into six groups which are shown in the figure. The individual labelled DOM is the domestic horse. The grey shaded individuals represent the horses for which we have phenotypic information.

already occurred in the captive population, the genetic variability has been reduced and one of the main aims of any breeding program is to maintain as much diversity as possible through successive generations.

There are four factors to be considered when trying to maximise the genetic diversity, and they usually all work in conjunction to reduce the genetic variability of captively managed populations.

Founder Effect

The number of founders of a captive breeding population limits the proportion of the total genetic variation present in the founder population when compared to the background populations from which they were taken. The proportion of genetic variation remaining in a small population is $1 - 1/2n$, where n is the number of founders (Nei et al. 1975).

Genetic Drift

Genetic drift is the term used for the random fluctuations in allele frequency from generation to generation and in a small population, the random drift could result in the loss of rare alleles.

Inbreeding

Inbreeding increases the probability that offspring will receive identical genes by descent and increases the proportion of homozygotes to heterozygotes, therefore increasing the expression of recessive traits; since most genes for deleterious traits are thought to be recessive, it increases the probability of inherited defects in a population.

Selection

Selection reduces genetic variability through elimination of alleles associated with genotypes which have been selected against either by artificial means (choosing the breeding stock) or naturally (susceptibility to disease). Whereas the latter may increase fitness through selection, selection on breeding stock on the basis of conformity to a hypothetical type reduces genetic variation and increases inbreeding in future generations.

All of these factors must be borne in mind when managing a captive population. However, by far the largest problem connected with the management of this pedigree is how best to incorporate the domestic genes from DOM which are carried by a high proportion, approximately $\frac{2}{3}$, of the population.

Breeding Policy in the United States

Breeding policy in the US before 1988 consisted of separating the population into two groups – those not descended (pure) and those descended (contaminated) from DOM. In the contaminated group, stallions were never bred from and the mares were bred with stallions from the pure group. This policy was introduced to reduce the frequency of domestic genes in the contaminated group gradually without reducing the growth rate of the population as a whole. The effect of the policy would be to eradicate all the DOM genes in the population, but at what cost? Geyer, Thompson, and Ryder (1989) showed that exclusion of the genes of DOM from the population would also mean the exclusion of genes from the founders 11 and 12 and would result in an average loss of 2.036 genes per autosomal locus. As a result of this, this group is now managed differently; the aim is to preserve its genetic diversity by no longer trying to eradicate the domestic genes. Owners of the pure group are reluctant to introduce domestic genes into the population and so the gene pool is maintained by restricting the gene flow into the pure population by breeding pure mares with pure stallions only. This approach assumes that only the domestic founding mare, DOM, brought domestic genes into the population, however, there existed anecdotal evidence that breeding between domestic and Przewalski horses

was widespread in the Mongolian plains implying that all of the founders would contain varying amounts of domestic genes. For example, it is now known that individual 18 is a hybrid of domestic and Przewalski horses (Dolan 1982).

Breeding Policy in Europe

In Europe, policy is to use a mild form of selection. This appears to contravene one of the basic rules of captive breeding which is to avoid selective breeding as in general, with the exception of some lethal genes, humans cannot determine which phenotypic characteristics are of advantage in either captive or wild environments. Even when we can determine the selective advantages, these are restricted to today's environment and as environment often changes rapidly and irregularly, what is of advantage today might be deleterious tomorrow and vice versa.

Every wild population has individuals that carry 'bad' genes but the same animals might also carry 'good' genes that have selective advantages. Selection acts not simply on one gene but on the entire genome, and could result in the loss of rare 'good' wild alleles. Severe human selection in combination with inbreeding has already occurred in this captive population, and will have resulted in 'new' phenotypes that are not observed in the wild and may have caused the homozygous expression of rare alleles that are only usually seen in domestic horse breeds. On the other hand some of these 'new' phenotypes may have originated from the domestic mare and the other founders who are regarded as not pure bred. These domestic phenotypes, for example a white star on the forehead, hanging manes and black nose, are considered undesirable by many people and a mild form of selection against these characteristics was initiated.

The 'fox' allele is an allele that gives the horse a red-brown colouring. It is considered to be a single recessive trait and a rare wild allele rather than a domestic allele whose numbers have risen sharply in the population. Princee (1990) compares five selection models for the 'fox' allele ranging from non selection through to removal of all possible carriers with an arbitrary probability greater than 0.25 and their effect on genetic variation. He concludes that a mild form of selection involving removal from the breeding stock of either true male, or both true male and true female, carriers has negligible effect on the variation of the population with the removal of only six and nineteen individuals respectively. The adopted policy is to avoid breeding from animals displaying the 'fox' characteristic and, whenever possible, the individuals of pairs that produce 'foxes' should be bred with non carriers. Recommendations are made for selection against the domestic characteristics. Selection is not recommended against horses that show only a single domestic feature but should be performed against horses that show several characteristics simultaneously. By doing this form of mild selection it is hoped to maintain these genes as part of the gene pool but to keep their frequency within reasonable limits.

Aims

We are going to use the pedigree to provide probabilities on the founding allele frequencies which should help us answer the questions about the different management programs. To do this we break the pedigree up into the six groups shown in Figure 2-1. By calculating the group allele frequencies we should be able to see whether the differences in the breeding programs are justified by seeing if there are any differences in the frequencies between groups containing the domestic horse (DOM), the suspected hybrid 18 and the remaining 'pure' Przewalski horses. We should also see if the horse 231, captured in a different location and introduced later into the pedigree than the original founders, shows any different frequencies.

2.2 Likelihood

Of the sixteen polymorphisms, ten have two alleles, three have three alleles and one each have four, five and six alleles. For reasons of computational intensity, outlined below, here we will only consider the two and three allele markers; however, in Chapter 3 we will expand this technique to deal with the higher allele traits.

If we are going to use the pedigree to provide calculations to resolve the management questions of breeding policy and interest focused on the three founders 18, 231 and DOM, it is important that we only consider the founders, as it is their genes and their genes only that contribute to the gene pool. The current allele frequencies may be subject to genetic drift and consequently may give misleading ancestral inference.

As shown in Equation (1.3), we base calculations on

$$L\{\phi(D), G, M\} = P\{\phi(D)|G, M\} \tag{2.1}$$

but in Equation (1.4) the likelihood was broken down into penetrance, transmission and population frequencies. We are not interested in changing the model for penetrance or transmission so we can re-write to give

$$L\{\phi(D), \pi_F\} = P\{\phi(D)|\pi_F, G\}. \tag{2.2}$$

This means that the likelihood of the ancestral genotype is equal to the probability of the ancestral genotypes given the founder population frequencies, denoted π_F, and the pedigree structure. By inputting the allele frequencies into Equation (2.2) and running the peeling algorithm on the PH pedigree we can obtain the likelihood for that particular frequency. So, by running the program many times with different allele frequencies we can get the likelihood curves for each trait.

The peeling method is an efficient algorithm for calculations of this sort on complex pedigrees but it is still an intensive computation, the requirements of which increase rapidly with the number of alleles at a locus, and we must do all that we can to reduce the memory requirements. In order to obtain the likelihood we only need to peel back to one individual.

This reduces memory requirements and means that we could technically find a maximum cutset of one to reduce the computational cost. Finding the optimum sequence of cutsets to give a maximum cutset of one using simulated annealing to reorder the summing out of individuals is virtually impossible, but finding a maximum cutset of about four or five reduces each likelihood calculation to a run time of only several seconds on a SPARC Server 1000. However, despite all of these memory savings, for this set of the Przewalski horse pedigree only the two and three allele traits can be peeled which leaves three traits unconsidered.

2.2.1 The Two Allele Traits

Likelihood Curves

We use this process to obtain the log likelihood curves for the two allele traits. Figure 2-2 gives the log likelihood curves for allele 1 for each of the two allele traits. They were obtained by using 20 different allele frequencies.

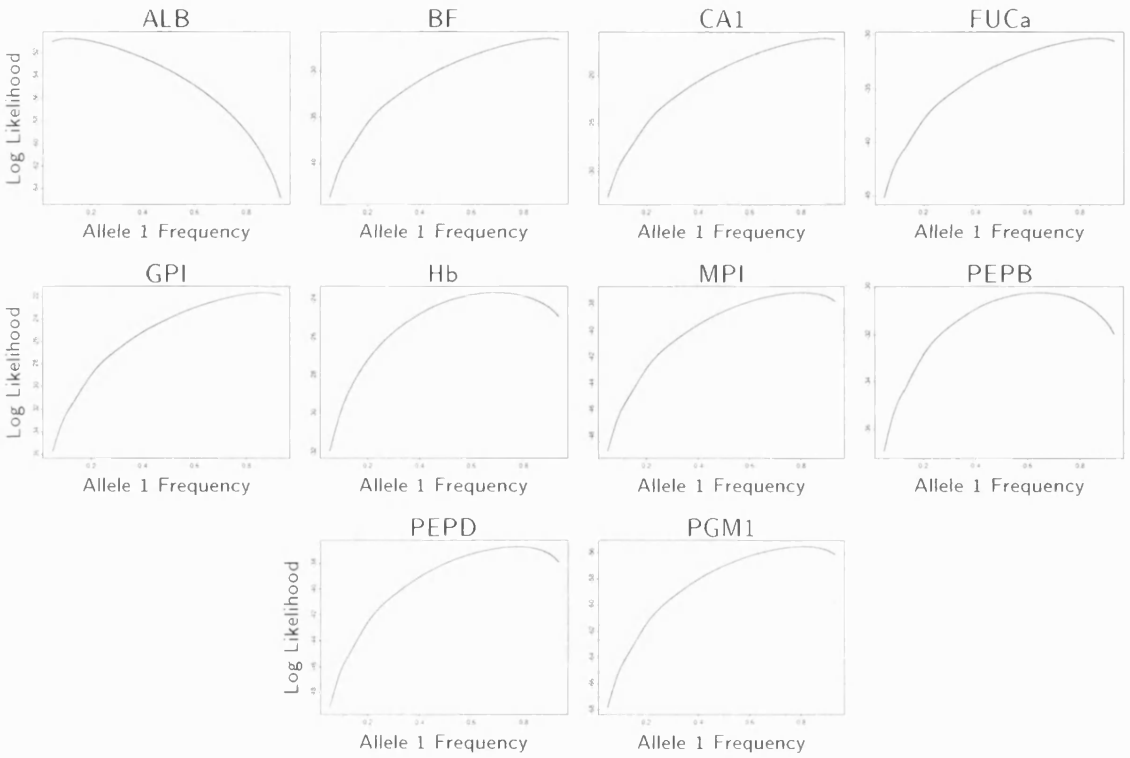


Figure 2-2: Log-Likelihood Curves for the Two Allele Traits: The figure shows the likelihood curves for allele 1 for each of the two allele traits. The curves are obtained by running the peeling algorithm for 20 different allele frequencies.

Care has to be taken when comparing the curves as the scales on the log-likelihood axes are not the same for each graph. The likelihood surface drops away suddenly as you approach the boundaries of the sample space and this is why the vectors of allele frequencies used to calculate the log-likelihood on the graphs are > 0.05 and < 0.95.

Most of the log-likelihood curves are skewed, typically maximum values are between 0.8 and 0.9 – or 0.1 and 0.2 – with only traits Hb and PEPB showing more centred curves. Of the skewed curves only one, ALB, is skewed to the right. This means that it is the only trait to have its allele 1 as its rarer allele.

Maximum Likelihood Estimates

In order to find the MLEs of the founding allele frequencies we have to use the peeling process repeatedly. By running the process over 100 points from 0 to 1 we obtain estimates of the MLEs to two decimal places. For example using trait ALB, the MLE of the rarer allele frequency was found to lie between 0.11 and 0.12, for further accuracy we have to run the process again using another 100 points but this time between 0.11 and 0.12.

The results of the MLEs for the rarer allele for each of the two allele traits using this laborious process are given in Table 2.1. The run time for all of the traits are between 2–10 seconds per likelihood calculation which equates to a total run times of approximately 1–3 hours for the 1000 likelihood calculations. Also shown in Table 2.1 are the approximate 95% confidence intervals for the MLEs. These are obtained by inverting a hypothesis test. If $H_0; \theta = \theta_0$ vs $\theta \neq \theta_0$ then approximately under H_0

$$2\{l(y;\theta_0) - l(y;\hat{\theta})\} \sim \chi^2_1. \tag{2.3}$$

To reject H_0 at the 95% level then

$$l(y;\theta_0) - l(y;\hat{\theta}) > 1.96^2/2 \approx 2. \tag{2.4}$$

We know the ML values for the allele frequency for each trait and so we find the allele frequencies that reduce the loglikelihood by ≈ 2 to give the 95% CIs.

The Rarer Allele MLE of the Two Allele Traits using a Decimal Search Method					
	Trait				
	ALB	BF	CAI	FUCa	GPI
Mle	0.1131	0.1066	0.1094	0.1341	0.1298
95% CI	(0.0062)	(0.0059)	(0.0060)	(0.0076)	(0.0073)
	(0.4307)	(0.4118)	(0.4144)	(0.4725)	(0.4727)
	Trait				
	Hb	MPI	PEPB	PEPD	PGM1
Mle	0.3062	0.1977	0.3574	0.2264	0.1947
95% CI	(0.0300)	(0.0165)	(0.0537)	(0.0393)	(0.0155)
	(0.6940)	(0.5611)	(0.7532)	(0.5692)	(0.5543)

Table 2.1: MLEs of the Two Allele Traits: The table shows the MLEs and their associated 95% CIs of the rarer allele for each of the two allele traits. The MLEs are obtained using repeated use of the peeling algorithm over a finer and finer grid of allele frequencies and the 95% CIs are obtained by finding the points that give a drop of ≈ 2 on the loglikelihood scale from the MLE.

From the results we can see that the confidence intervals are very wide indicating that drawing inferences about the founding population will be difficult.

2.2.2 The Three Allele Traits

The same method is used to obtain the likelihood for the three allele traits. This time we are varying two parameters and the likelihood is a surface rather than a curve. However, as the sum of allele frequencies of the three alleles cannot be larger than unity, the likelihood surface can only be obtained for the simplex bounded by the axes and the line $x + y = 1$. Outside this simplex the likelihood is zero.

In the last section, we indicated how laborious it is to find the MLEs for the two allele traits. The process to find the MLEs for the three allele traits is identically laborious but even more time consuming. The peeling algorithm has to be run initially on a grid of 100 by 100 allele frequencies to obtain the likelihood surface. By running on a grid of 100 by 100 points we only have accuracy of the MLE to two decimal places. In order to increase accuracy we have to find the best estimate and then run another 100 by 100 grid on a smaller sample space around the first estimate. For example, the first 100 by 100 grid gave (0.26, 0.45) as the MLEs for trait C71; we then re-run the whole process but concentrate on the square (0.26, 0.45), (0.26, 0.46), (0.27, 0.46) and (0.27, 0.45) which gives accuracy to four decimal places. Using this decimal search method, the MLEs for C71 and the other three allele traits are shown in Table 2.2.

MLEs of the Three Allele Traits using a Decimal Search Method			
	Trait		
Allele	C71	C72	Pa
1	0.2687 (0.0122,0.8203)	0.4988	0.5435
2	0.4510 (0.0345,0.9417)	0.2826	0.3542
3	0.2803 (0.0137,0.8411)	0.2186	0.1023

Table 2.2: MLEs of the Three Allele Traits: The table shows the MLEs of the allele frequencies for each of the three allele traits. Also shown in brackets are the 95% CIs for the MLEs for the C71 trait. The MLEs are obtained using repeated use of the peeling algorithm over a finer and finer grid of allele frequencies and the CIs are obtained from a drop of ≈ 2 on the loglikelihood scale.

Also shown in the table is the approximate 95% CIs for the allele frequencies for trait C71. These were obtained by using the similar method outlined above *i.e.*, reducing the loglikelihood by ≈ 2. This time due the increase in dimension, the interval is represented by an elliptical shape. The approximate intervals are obtained from the minimum and maximum values for each of the alleles around this elliptical shape. The results again show that the intervals are large making inference difficult. The results for the other traits are similar but not shown.

For traits C71 and Pa the cutset cost was 5 and 6 and the likelihood calculation took a few seconds for each grid point. The cutset for C72 was 7 and so each calculation took approximately 10 seconds. This represents a total run-time for accuracy to four decimal places of approximately 18 hours for traits C71 and Pa and approximately two days for trait C72. We note however that as the plots look quadratic there are algorithms, such as NAG routines and variable metric methods that would be much more efficient at finding the maximum.

Figures 2-3 are the plots of the likelihood surface for each of the three allele traits obtained from a 100×100 grid of allele frequencies. The values of the likelihood are scaled by dividing by the MLEs so the scale on the figures represent the proportion of the MLEs. The 1 on the figures represents the MLE.

2.3 The EM Algorithm

So we have a procedure to find the MLEs but the process is very slow and laborious. Can we do any better? From Equation (2.2) we note that if we are given the ancestral genotypes, the founder population allele frequencies can be estimated by a simple gene count. Further, if we are given the founder allele frequencies, the probabilities of each ancestor's genotype can be calculated using the peeling method. By iterating between the gene-counting and peeling steps we can obtain the MLEs of founding allele frequencies and ancestral genotype probabilities.

In fact, this is an EM algorithm (Dempster, Laird, and Rubin 1977) with the Estimation step consisting of obtaining the ancestral genotype probabilities via the peeling process and the Maximisation step amounting to the calculation of the new population frequencies from these probabilities. This data can be thought of as counts and not real data values, however, in §1.5 we showed that handling categorical data in this way fits into the EM framework.

We need to provide the algorithm with initial values. The estimate of the current population frequencies obtained by gene counting on the observed phenotypes provides a good point as these estimates should be close to the founding allele frequencies. We can also compare the frequencies of the ancestral and current populations although the data observed on our pedigree are not an independent sample.

2.3.1 The Two Allele Traits

The above EM algorithm was implemented for each of the two allele traits. As we have previously mentioned, memory restrictions are the main concern when running the peeling process. Unlike the likelihood calculations where we could peel back to one individual, this time we have to peel back to all 13 founders to enable us to perform the maximisation step of a gene count on the founding population, and so it appears that we have to run the program 13 times. However, we don't need the joint distribution we just need the expected frequency of copies of the genes and so for computational efficiency it is optimal

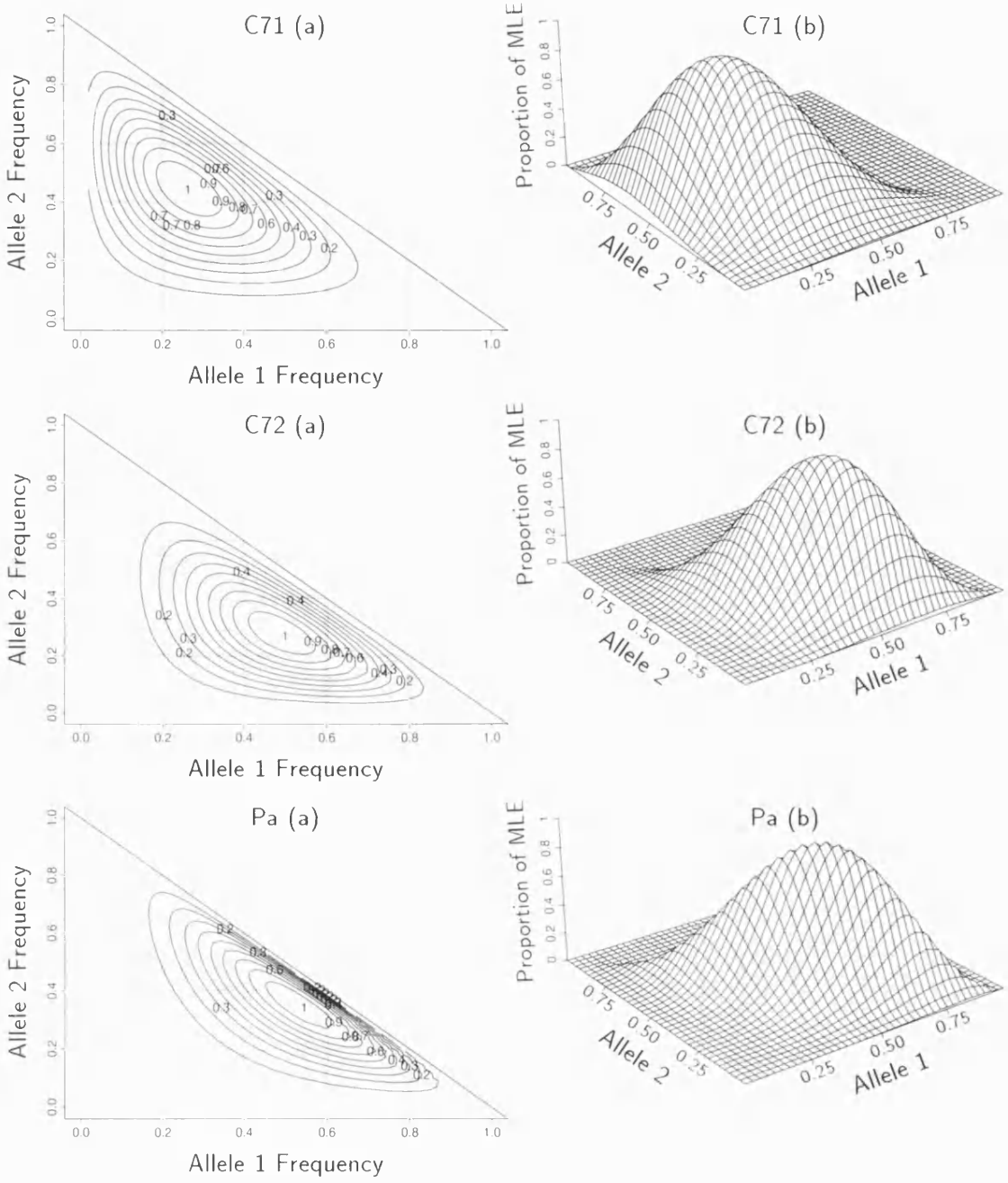


Figure 2-3: Likelihood Surface for the Three Allele Traits: Figures (a) show the contour and Figures (b) show the perspective plots of the exact likelihood surface for each of the three allele traits. The surface was obtained by the peeling algorithm for a grid of 100 by 100 allele frequencies. The Maximum Likelihood is shown with a 1. The grey shaded lines are for reference only.

to split the 13 founders into some form of groups.

Prepeeling

As illustrated in §1.6, we use the fact that a child of a marriage in the pedigree offers no further genetic information so we can effectively 'clip' the individual out of the pedigree by collapsing its genetic information onto the parents. We use a prepeeling program for each trait to reduce the number of individuals in the pedigree from 244 to approximately 170.

Optimisation

Also illustrated in §1.6 is that the order of summing out of an individual is important. If the maximum cutset is too large, the peeling algorithm will fail due the large amounts of storage required and this increases with the number of alleles in the trait.

Ideally, the cost of the maximum cutset is the number of individuals in the cutset but the optimal ordering giving this minimum cost is rarely obtained. We have a large number of possible orderings and some cost function associated with each one. We use the combinatorial optimisation procedure, simulated annealing §1.12.3, to search over the set of possible orderings minimising the cost function.

The cost function is the log of the total storage requirement at any time. This seems a slightly odd way of defining the cost function but the storage requirement is proportional to the size of the cutset and unfortunately, you can have several independent cutsets that you need to store functions on at the same time. Suppose that these cutsets have sizes $s_1, s_2, ...s_k$ and assume that the number of genotypes you are considering is g then the total storage requirement is

$$T = g^{s_1} + g^{s_2} + ... + g^{s_k}. \tag{2.5}$$

The quantity that gets minimised is $log_g(T)$. This function is chosen because T is what you really want to minimise and taking log_g means that if you have one cutset of size 5 say, the cost is 5. As soon as we have obtained a maximal storage requirement and hence workable maximum cutset size we then store this order of summation and use repeatedly.

We have stated above that rather than doing each calculation for each of the founders we can group the horses in some way. However, we are dealing with 13 individuals and even for these two allele traits, the minimum cutset cost of 13 for running the program once for all of the founders would be too large, so the set of founders has been broken down into the groups illustrated in the Przewalski horse pedigree, Figure 2-1. These groups were chosen as an almost identical summation order would be required for the individuals in each group, resulting in an increase in efficiency. This makes the cutset cost smaller and makes the calculation possible, but it makes the whole process rather inefficient, as the algorithm has to do almost identical calculations several times.

For the simulated annealing algorithm we typically require 50,000 iterations using a

starting temperature of 10 and a geometric cooling schedule with temperature parameter 0.997 to obtain workable cutsets. We use 10 EM iterations with the initial frequencies equal to the current population estimates. A SUN SPARC 1000 and the `Pedpack` (Thomas 1991) pedigree package were used to run the program. Using the previously found optional ordering, each iteration of the peeling process took approximately two minutes.

Results

In the first column of Table 2.3, the MLEs obtained from the exhaustive search method are shown, along with the 95% CIs obtained from the method of inverting a hypothesis test and finding the values that gives a drop of ≈ 2 on the likelihood scale. Also shown is the standard deviation of the allele frequencies of the founding individuals. The second column shows the estimates obtained from running 10 EM iterations. The third column displays a different estimate – the current allele frequencies – and is obtained from a gene count on the observed data. The first two columns are estimating the same thing and so should be close, where column 3 is estimating using a different route and so should be different. The column n represents the number of individuals for which we have phenotypic information for each trait. Only the frequency of the rarest allele in the founding population is displayed and results that are different by approximately 0.1 are shown in bold.

Current and Founding Populations

As we expected, there are the substantial differences between the first two columns and column 3 of the table, the founding and current allele frequency estimates. Only two out of the ten traits, ALB and FUCa, have the rarest allele frequency increasing from the founding population to the current population; FUCa shows a large increase of 0.1556, whereas ALB shows only a small increase of 0.0046. The remaining eight traits show decreasing rarer allele frequencies, with the most notable decreases being PEPD which shows a drop of approximately 0.17; MPI and CA1 which show drops of approximately 0.1. Due to the drop in the rarer allele in traits CA1, GPI and PEPD, these alleles are in danger of disappearing altogether.

If we compare the founding allele frequencies obtained from only 10 EM iterations with the MLEs obtained from a decimal search technique, we can see that using only 10 peeling EM iterations has found the MLEs to almost three decimal places. The peeling EM algorithm used initial starting values of the current allele frequencies and so where there is only a small difference in the current and founding population estimates, for example traits ALB and BF, 10 EM iterations gets us very close to the MLEs; when there is a larger discrepancy between the current and founding estimates, *eg* trait PEPD, then the estimate is not quite so exact and more EM iterations are required.

The Population Allele Frequency Estimates					
			Founding Pop		Current Pop
Trait	n		(Decimal Search)	(EM Algorithm)	(Gene Count)
ALB	126	Mle	0.1131	0.1133	0.1177
		95% CI	(0.0062,0.4307)		
		SD	(0.0832)		
BF	135	Mle	0.1066	0.1063	0.1037
		95% CI	(0.0059,0.4118)		
		SD	(0.0660)		
CA1	124	Mle	0.1094	0.1090	**0.0161**
		95% CI	(0.0060,0.4144)		
		SD	(0.0492)		
FUCa	126	Mle	0.1341	0.1350	**0.2897**
		95% CI	(0.0076,0.4725)		
		SD	(0.0469)		
GPI	128	Mle	0.1298	0.1293	**0.0586**
		95% CI	(0.0073,0.4727)		
		SD	(0.0799)		
Hb	128	Mle	0.3062	0.3056	0.2539
		95% CI	(0.0300,0.6940)		
		SD	(0.0767)		
MPI	123	Mle	0.1977	0.1972	**0.1219**
		95% CI	(0.0165,0.5611)		
		SD	(0.0670)		
PEPB	124	Mle	0.3574	0.3573	0.3398
		95% CI	(0.0537,0.7532)		
		SD	(0.1197)		
PEPD	128	Mle	0.2264	0.2252	**0.0547**
		95% CI	(0.0393,0.5692)		
		SD	(0.0748)		
PGM1	128	Mle	0.1947	0.1945	0.1602
		95% CI	(0.0155,0.5543)		
		SD	(0.0578)		

Table 2.3: Population Allele Frequency Estimates: The population allele frequency estimates for the rarer allele of the two allele traits are shown for three different methods. The current population estimates are obtained from a gene count on the individuals on which phenotypic information is available. The founding population estimates are for 10 peeling EM iterations using the current population estimates as the initial values. The MLEs obtained from the decimal search method in §2.2.1 are shown for comparison. Results that are different by approximately 0.1 are shown in bold.

Management Issues

Having obtained accurate estimates of the founding frequencies via the peeling EM algorithm we then use these in a further iteration of the peeling program to find the allele frequencies of the six groups shown in Figure 2-1. Again, Table 2.4 shows only the frequency of one allele, the rarest overall allele in the founding population. The number of individuals for which we have phenotypic information for each trait is again shown and results that are different by about 0.1 are shown in bold. To try and give some notion of variability, the table also shows the within-group standard deviation, that is the standard deviation of the allele frequencies of the horses in the group. Due to the pedigree, both individuals in group 2a have identical expected numbers of gene copies giving the standard deviation of 0. The table also shows the approximate 95% CIs for the groups; these were obtained by running the peeling program using the 95% CI points for the MLE of the allele frequencies and shown in Table 2.3.

Individual 231, who is in group 5, has very few entries in Table 2.4. She has been prepeeled out of the pedigree for many of the traits because she is towards the bottom of the pedigree and produced only one line of heredity.

The effect of any unusual allele proportions for the hybrid 18 may be diluted, due to the fact that group 2 contains four founders, see Figure 2-1, so because of the way the group divides, group 2 has been split into two sub-groups 2a and 2b. Group 2a contains the founders 17 and 18 and 2b contains the founders 39 and 40. The average of these two sub-groups will clearly give the frequencies for group 2 but the differences between the two sub-groups are of interest.

Group Differences

We indicated that due to the large CIs for the MLEs, inference about individual groups could be difficult. We can see that this is the case when we examine the 95% CIs of each group. The CIs are large and hence overlap any discrepancies in the group estimates and so to make inference about any particular group, we must look if any group continually stands out with unusual allele frequencies.

Groups 2a and 2b show no extreme results and exhibit no trend. These sub-groups do not show widely differing frequencies when compared to the rest of the groups and the proportions are quite close with the largest difference of 0.1236 occurring in the MPI trait.

Group 5 contains only the single horse captured in a different area from the remaining founders. In many of the traits, the best estimate of this horse's frequencies are the founding population frequencies as 231 has been prepeeled out of the pedigree. Her allele frequency estimates for two of the two allele traits, ALB and PEPB, show no extreme results. However, the result for trait Hb is extreme with the rarer allele just becoming the more frequent at 0.5080.

Group 1 is the group that contains the domestic mare. The group only shows a striking result in one of the markers, BF, otherwise it appears no more unusual than the

Allele Frequency Estimates of the Six Founding Groups Obtained Using Founding Population Frequencies								
			Founding Group					
Trait	n		1	2a	2b	3	4	5
ALB	126	Mle	0.0748	0.0851	0.0599	0.0768	**0.2488**	-
		SD	0.0134	0	0.0044	0.0240	0.0546	-
		95% CI	(0.0041)	(0.0046)	(0.0032)	(0.0157)	(0.1634)	-
		Values	(0.2976)	(0.3360)	(0.2441)	(0.2681)	(0.5073)	-
BF	135	Mle	**0.2167**	0.0939	0.0635	0.0610	0.0837	-
		SD	0.0231	0	0.0052	0.0283	0.0092	-
		95% CI	(0.1561)	(0.0171)	(0.0114)	(0.0033)	(0.0046)	-
		Values	(0.4071)	(0.3367)	(0.2361)	(0.2452)	(0.3289)	-
CA1	124	Mle	0.1877	0.1207	0.0838	0.0633	0.0884	-
		SD	0.0116	0	0.0063	0.0287	0.0086	-
		95% CI	(0.1250)	(0.0445)	(0.0308)	(0.0034)	(0.0048)	-
		Values	(0.3806)	(0.3553	(0.2540)	(0.2496)	(0.3408)	-
FUCa	126	Mle	0.1891	0.1857	0.1283	0.0778	0.1085	-
		SD	0.0107	0	0.0096	0.0353	0.0105	-
		95% CI	(0.0923)	(0.0779)	(0.0539)	(0.0044)	(0.0061)	-
		Values	(0.4321)	(0.4621)	(0.3265)	(0.2873)	(0.3903)	-
GPI	128	Mle	0.0868	0.1066	0.1015	**0.2959**	0.1188	-
		SD	0.0153	0	0.0013	0.0797	0.0048	-
		95% CI	(0.0048)	(0.0060)	(0.0124)	(0.2459)	(0.0067)	-
		Values	(0.3315)	(0.3920)	(0.3405)	(0.4498)	(0.4317)	-
Hb	128	Mle	0.2077	0.2415	0.3023	0.2885	0.3948	**0.5080**
		SD	0.0366	0	0.0100	0.0202	0.0403	-
		95% CI	(0.0197)	(0.0246)	(0.0279)	(0.2568)	(0.0431)	(0.0649)
		Values	(0.5064)	(0.5614)	(0.5821)	(0.4847)	(0.7497)	(0.8004)
MPI	123	Mle	0.1322	0.1490	0.2726	0.2046	0.2746	-
		SD	0.0235	0	0.0225	0.0051	0.0313	-
		95% CI	(0.0108)	(0.0123)	(0.0421)	(0.2271)	(0.0369)	-
		Values	(0.3969)	(0.4392)	(0.5374)	(0.3812)	(0.6068)	-
PEPB	124	Mle	0.2442	0.2839	0.3741	**0.5580**	0.3751	0.3555
		SD	0.0429	0	0.016	0.1498	0.0078	-
		95% CI	(0.0353)	(0.0432)	(0.1051)	(0.3164)	(0.0698)	(0.0574)
		Values	(0.5589)	(0.6171)	(0.6329)	(0.7849)	(0.7390)	(0.7283)
PEPD	128	Mle	0.2456	0.2190	0.1675	0.1538	**0.3375**	-
		SD	0.0073	0	0.0092	0.0487	0.0467	-
		95% CI	(0.1288)	(0.0807)	(0.0623)	(0.0455)	(0.1836)	-
		Values	(0.4746)	(0.4897)	(0.3764)	(0.3689)	(0.6210)	-
PGM1	128	Mle	0.1339	0.1562	0.2541	0.2124	0.2607	-
		SD	0.0224	0	0.0175	0.0006	0.0253	-
		95% CI	(0.0106)	(0.0127)	(0.0359)	(0.0231)	(0.0322)	-
		Values	(0.3990)	(0.4521)	(0.5051)	(0.3836)	(0.5855)	-

Table 2.4: Allele Frequency Estimates for the Six Founding Groups: The allele frequency estimates of the six founding groups in the pedigree indicated in Figure 2-1 are shown. They are obtained by running a peeling step using the founding allele estimates.

other groups.

The Three Interesting Founders

Over the years anecdotal evidence, such as photographs (Dolan 1982, Bouman 1982), has suggested a large diversity in the founders and this is the first thing that is apparent from these results. The differences in the allele frequencies between the groups are considerable even when taking into account the possible corruption of the two domestically linked horses and the animal from a different geographical location. Even groups 3 and 4 which both contain Przewalski horses captured at the same time in the same location show some large differences in their allele frequencies.

It has been known for some time that matings between Przewalski's and domestic species can occur and result in fertile offspring. Hence, the two breeds could have a large number of genes in common and the results from the analyses concerning the two known carriers of domestic genes support this. In the case of the hybrid, founder 18, a look at the sub-groups 2a and 2b shows that they are not extreme and the differences between them are quite small when compared to the rest of the group frequencies. In group 4, the group which contains DOM, only two results in the ALB and PEPD traits are extreme. So in both cases little differences between the allele frequencies of the two breeds can be found.

The individual 231 shows a large discrepancy in its allele proportions for Hb. She was not one of the known domestically linked horses but the horse from a different geographical location. Conclusions on this individual are tempered by the fact that for seven of the loci there is no information for 231.

All of this means that we can make no strong conclusions about the groups.

2.3.2 The Three Allele Traits

The EM algorithm outlined in §2.3 can also be used on traits with three alleles.

Optimisation

The order of summing out of individuals is even more important here than with the two allele traits. We again have to split the 13 founders into smaller groups of three or four to ensure that the combinatorial optimisation algorithm, simulated annealing (SA), can reduce the maximal cutset to a manageable cost of about 7. If the cost is larger then the memory of the machine is insufficient to complete the calculation. In order to achieve this cutset cost, a lot of annealing iterations are required with a small cooling parameter. As in the case of the 2 allele traits, as soon as we have obtained a workable cutset via the SA algorithm, we store the order of summing and use that order repeatedly.

Results

We have found that we require 500,000 iterations of the SA algorithm using a geometric cooling schedule with temperature parameter 0.9997. The run time for each iteration of the peeling EM algorithm is longer and more variable than for the two allele calculations because of the increase in alleles, and the maximum cutset cost varies for each trait. The cutset cost for traits C71 and Pa is 5, but the the cost for C72 is 7 and this trait needs several runs of the SA algorithm with different starting temperatures to obtain a workable maximum cutset size.

Again we run 10 iterations of the EM algorithm using the current population estimates as initial values. Each EM iteration takes between three and ten minutes depending on the cutset cost for the trait.

Table 2.5 shows estimates of the founding and current population. The founding population estimates are obtained via a decimal search method and 10 iterations of the peeling EM algorithm. Column 3 of the table gives the current population estimates obtained via a gene count on observed data. Also shown are the 95% CIs for the MLEs for trait C71 from Table 2.2. All the allele frequencies are displayed this time in order to make comparisons.

The Population Allele Frequency Estimates						
				Founding Pop		Current Pop
Trait	n	Allele		(Dec Search)	(EM Algr)	(Gene Count)
C71	63	1	Mle	0.2689	0.2683	0.3095
			95% CI	(0.0122,0.8203)		
		2	Mle	0.4510	0.4507	0.3730
			95% CI	(0.0345,0.9417)		
		3	Mle	0.2801	0.2809	0.3175
			95% CI	(0.0137,0.8411)		
C72	49	1		0.4988	0.4995	0.5102
		2		0.2826	0.2830	0.3571
		3		0.2186	0.2175	**0.1327**
Pa	138	1		0.5435	0.5431	**0.6377**
		2		0.3542	0.3545	**0.2862**
		3		0.1023	0.1024	0.0761

Table 2.5: Population Allele Frequency Estimates: The population allele frequency estimates for the three allele traits are shown for three different methods. The current population estimates are obtained from a gene count on the individuals on which phenotypic information is available. The founding population estimates are for 10 EM iterations using the current population estimates as the initial values. The MLEs obtained from the decimal search method in §2.2.2 are shown for comparison. Results that differ by approximately 0.1 are shown in bold.

Current and Founding Populations

In the previous section, we showed that the peeling EM algorithm rapidly obtains the MLEs of the founding allele frequencies for the two allele traits. For the three allele traits, the founding population allele frequency estimates, again obtained from 10 EM iterations using the current population estimates to start the algorithm, are close to the MLEs found in §2.2.2 by the painstaking decimal search method. In fact, using only 10 EM iterations they are nearly correct to three decimal places and use far less computational time.

Comparing the founding and current population estimates we can see that there are large differences. C72 shows a large loss of approximately 0.085 in the rarest allele frequency and Pa also shows a small loss, but also of note in this marker is the gain of frequency in the commonest allele at the expense of the second commonest allele. Interestingly, trait C71 shows a change almost to parity between the three alleles.

Management Issues

To answer the management questions of interest we then run the peeling algorithm using the founding population estimates and find the allele frequency estimates for each of the founding groups indicated in Figure 2-1. The results are shown in Table 2.6.

Allele Frequency Estimates of the Six Founding Groups Obtained Using Founding Population Frequencies								
			Founding Group					
Trait	n	Allele	1	2a	2b	3	4	5
C71	63	1	0.1991	0.2287	0.2199	0.4871	0.3843	**0.1250**
		2	0.5253	0.5217	0.6107	0.3905	0.4487	**0.2404**
		3	0.2756	0.2496	0.1694	0.1225	0.1670	**0.6346**
C72	49	1	0.5613	0.4407	0.4447	0.4325	0.4879	-
		2	0.2585	0.3051	0.2570	0.3304	0.3119	-
		3	0.1803	0.2542	0.2983	0.2370	0.2002	-
Pa	138	1	0.4683	0.5968	0.6285	0.4994	0.4845	-
		2	0.4657	0.3280	0.3205	0.4401	0.2847	-
		3	0.0660	0.0752	0.0510	0.0605	**0.2308**	-

Table 2.6: Allele Frequency Estimates for the Six Founding Groups: The allele frequency estimates of the six founding groups in the pedigree indicated in Figure 2-1 are shown. They are obtained by running a peeling step using the founding allele estimates. The column n represents the number of individuals for which we have phenotypic information. No results were possible for individual 231 (founder group 5) for two traits as she has been prepeeled out of the pedigree. Results that differ by approximately 0.1 are shown in bold.

Group Differences

Groups 2a and 2b show no extreme results.

The results for group 5, the horse captured in a different area from the remaining founders, can only be obtained for one of the three allele traits. The results for C71

indicate large differences in the frequencies for allele 2 and allele 3 of approximately 0.15 and 0.35 respectively when compared with the other groups.

Group 1 appears no more unusual than the other groups.

The Three Interesting Founders

Apart from the large differences displayed by 231, group 5, for trait C71, the only other striking result is for the Pa allele 3 for group 4 which doesn't contain any of the three interesting founders. So groups containing the three interesting founders appear to be no different to the other groups due to the large differences between the founding allele frequencies of the groups.

2.4 Sensitivity Analysis

A lot of time has been spent finding the MLEs of the founding population for each trait, but how important is it that they are used for ancestral inference? If we run the peeling algorithm using the current population frequencies instead of the founding population frequencies to calculate the founding and group frequencies we get the results shown in Tables 2.7 and 2.8.

The Population Allele Frequency Estimates					
Trait	n	Allele	MLE (Dec Search)	Founding Pop	Current Pop (Gene Count)
C71	63	1	0.2687	0.3133	0.3095
		2	0.4510	0.4834	0.3730
		3	0.2803	0.2033	0.3175
C72	49	1	0.4988	0.4211	0.5102
		2	0.2826	0.3281	0.3571
		3	0.2186	0.2508	**0.1327**
Pa	138	1	0.5435	0.4658	**0.6377**
		2	0.3542	0.4203	**0.2862**
		3	0.1023	0.1139	0.0761

Table 2.7: Population Allele Frequency Estimates: The results for the allele frequencies estimates for the founding population are shown using the current population estimates. The current population uses a gene count method to estimate the frequencies and then runs a peeling step on them to obtain the founding population. Results that differ by approximately 0.1 are shown in bold.

If we compare these results with those in Tables 2.5 and 2.6 we can see substantial changes in both the founding estimates and the group estimates. In the founding population estimates, trait Pa shows a swing of approximately 0.08 between its allele 1 and allele 2 frequencies; when using the estimates of the founding population frequencies from the EM algorithm to we obtained estimates of (0.5431, 0.3545), whereas if we use the current population estimates we get (0.4658, 0.4208). In the group frequencies most show a shift,

Allele Frequency Estimates of the 6 Groups Obtained Using Founding Population Frequencies								
			Founding Group					
Trait	n	Allele	1	2a	2b	3	4	5
C71	63	1	0.2093	0.2404	0.2304	0.4968	0.3988	**0.1567**
		2	0.5114	0.5080	0.5994	0.3795	0.4320	0.3910
		3	0.2793	0.2516	0.1707	0.1238	0.1692	**0.4523**
C72	49	1	0.5021	0.3782	0.3790	0.3975	0.4122	-
		2	0.2927	0.3464	0.2948	0.3492	0.3595	-
		3	0.2052	0.3024	0.3207	0.2533	0.2283	-
Pa	138	1	0.4084	0.5268	0.5618	0.4634	0.4202	-
		2	0.5193	0.3877	0.3799	0.4690	0.3397	-
		3	0.0722	0.0855	0.0620	0.0676	**0.2401**	-

Table 2.8: Allele Frequency Estimates for the Six Founding Groups: The results for the allele frequencies estimates for the six founding groups indicated in Figure 2-1 are shown using the current population estimates. The current population uses a gene count method to estimate the frequencies and then runs a peeling step on them to obtain the allele frequency estimates for the six groups. Results that differ by approximately 0.1 are shown in bold.

the most dramatic being for group 5 with trait C71 which changes from (0.1250, 0.3910, 0.4523) to (0.1567, 0.2404, 0.6346).

These results show that for valid ancestral inference about the founders it is vital that we find and use the MLEs of the founding population frequencies.

2.5 Error Estimates

The peeling algorithm cannot provide us with standard errors and so we have no idea how uncertain the estimates obtained for the two and three allele traits are.

In order to get an estimate of the standard errors we assume that the allele frequency likelihood curves for the two allele traits are Gaussian. This implies that the log-likelihood curves shown in Figure 2-2 are quadratic. Fitting a quadratic, $y = a + bx - cx^2$, locally at the MLE and using a function of the x^2 term, we can obtain estimates of the variance from the $d^2y/dx^2 = -2c$ term. Figure 2-4 shows us quadratic spline approximation for two traits, ALB and Hb. In both cases three different quadratic splines were fitted. The MLE and two points equidistant from it - 0.01, 0.05 and 0.1 - were used to fit the spline. Care has to be taken interpreting any results as the scales on the log-likelihood axes are different.

The coefficients for each of the three spline fittings are given in Table 2.9. We are interested in the c coefficient and can see that the ALB c coefficient is larger than the Hb; this corresponds to a smaller standard error. In both cases, the estimate of c increases as the range increases. As, we are interested only in a local approximation for the curves we propose to use the smallest range, which corresponds to the quadratic fitting using the MLE and two points 0.01 equidistant. Table 2.10 gives the coefficients obtained for the

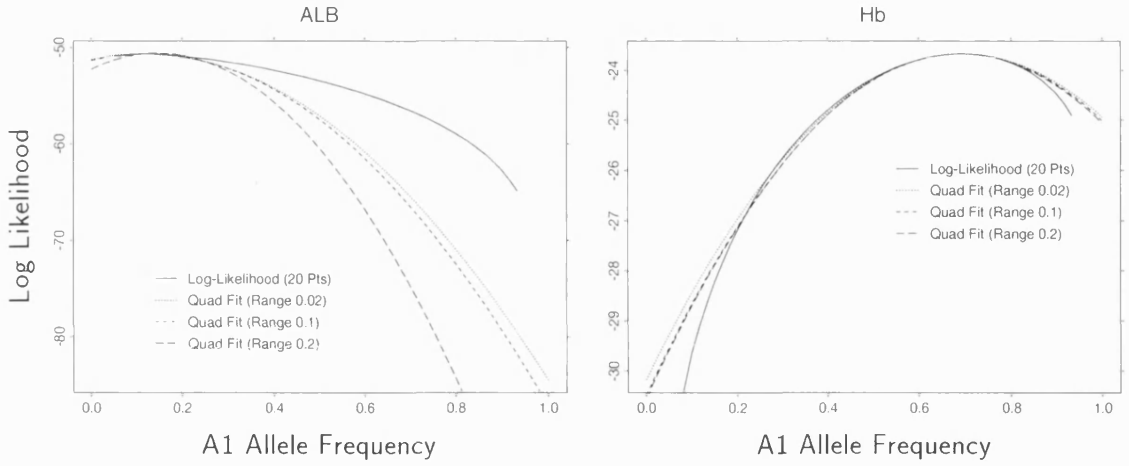


Figure 2-4: QUADRATIC SPLINES: Three quadratic splines are shown for the two allele traits ALB and Hb. The splines use the MLE and two points equidistant: $\pm 0.01, \pm 0.05$ and ± 0.1. The exact log-likelihood curves are shown for comparison.

QUADRATIC COEFFICIENTS						
	ALB COEFFICIENTS			Hb COEFFICIENTS		
RANGE	a	b	c	a	b	c
0.02	-51.24	9.77	-43.03	-30.18	18.74	-13.50
0.10	-51.36	11.31	-47.12	-30.48	19.65	-14.18
0.20	-52.25	22.71	-78.44	-30.55	19.93	-14.44

Table 2.9: QUADRATIC SPLINE COEFFICIENTS: The coefficients for quadratic fitting of the log-likelihood curves are given for traits ALB and Hb. Three different quadratic spline fitting approximations are used. Each spline uses the MLE and two points equidistant. The two additional points are $\pm 0.01, \pm 0.05$ and ± 0.10 respectively.

spline fitted using three points, the MLE and MLE ± 0.01, for each of the two allele traits. The subsequent standard error estimate is also shown.

Results

The estimates of the standard error seem large for all the traits with the smallest being 0.0876 for traits CA1 and BF and the largest being 0.1993 for PEPB. If we compare these estimates with the likelihood curves in Figure 2-2 we see that the smaller errors are associated with the curves which are most skewed and the larger errors, Hb and PEPB, are the most symmetric, which makes sense in view of the variance behaviour of the *Binomial* distribution.

Obtaining these estimates of the standard error allows us to check the 95% CIs obtained in Table 2.3. By multiplying the estimates of the standard error shown in Table 2.10 by 1.96 we get 95% CIs in the region of approximately ± 0.3. These are similar to the values got using the drop of ≈ 2 on the loglikelihood scale, but whereas that method takes account of the skewness of the distribution, this method takes no account of the skewness of the

Error Estimates for Quadratic Spline				
	Spline Coefficients			St. Error Est
Trait	a	b	c	$\sqrt{-1/2c}$
ALB	-51.24	9.77	-43.00	0.1078
BF	-77.98	115.87	-65.22	0.0876
CA1	-67.49	115.87	-65.17	0.0876
FUCa	-53.48	53.68	-31.13	0.1267
GPI	-44.43	52.56	-30.34	0.1283
Hb	-30.18	18.74	-13.50	0.1924
MPI	-53.15	40.09	-25.06	0.1413
PEPB	-35.43	16.17	-12.59	0.1993
PEPD	-53.50	43.84	-28.58	0.1323
PGM1	-71.51	39.70	-24.69	0.1423

Table 2.10: Error Estimates for the Two Allele Traits: Coefficients of the quadratic spline fitting using the MLE and two points ± 0.01 either side are shown for each of the two allele traits. Subsequent estimates of the standard error are also shown.

distribution and returns symmetric CIs.

In §4.3, we develop methods that enable us to obtain estimates of uncertainty. The uncertainty estimates are related to the marginal posterior means making comparison tricky. However, we find that this spline method appears to do well in obtaining estimates of standard errors.

2.6 Conclusions

The EM Method

It is vital that for ancestral inference we obtain the correct allele frequencies of the founders, as it is their genes and their genes only which make up the gene pool. We can then compare founding and current population frequencies to draw inferences as to the effect of the loss of rare alleles in the population.

The EM algorithm outlined in §2.3 provides us with accurate estimates of the MLEs of the founding allele frequencies. The main advantage is its speed. It obtains accuracy that would be lengthy and laborious to obtain by running the peeling algorithm over a grid of points on the sample space. We notice that after its first EM iteration, where the process appears to be affected by its starting frequencies. the iteration process then appears to move linearly to the MLEs. Plots of the peeling EM iterations are shown in Figure 3-1. The method could be further improved by using some kind of projection method to speed convergence. It would project forward along the straight line proportional to the size of the jump of the previous iteration and then use this new position from which to run the next EM iteration. The method is an uphill search and works well when the log-likelihoods are concave, as shown by all the traits in this chapter, but if the likelihood is more complicated then the algorithm might get stuck in a local maximum. This needs to be checked by running the EM algorithm several times from different starting points

which should pick up any multi-modality unless we are unfortunate with the choice of starting frequencies.

The Three Interesting Founders

Making inferences about the three interesting founders is very difficult due to the large 95% CIs around the MLEs for the founding allele frequencies. in order to make any inference we have to look whether any particular individual repeatedly stands out with unusual frequencies.

The individual which shows a large discrepancy in its allele proportions is not one of the known domestically linked horses but 231, the Przewalski horse from a different location. Conclusions on this individual are tempered by the fact that for nine of the loci there is no information for her and that she is the only horse in her group, but of the remaining three the results distinguish this horse from the remaining groups. Consequently, it would appear that the geographical location has a larger effect on allele frequencies than the domestic horses.

It is unwise to select on the grounds of ancestry as the animals are so tightly bonded. This analysis has found no real differences in the allele frequencies between the founder groups with domestic ancestry and Przewalski horses. This suggests a degree of inter-breeding between the species in the wild population.

Management Issues

The different breeding programs in the US and Europe are known as the *genotypic* and *phenotypic* approaches respectively. The aim of both is to preserve genetic diversity and the true natural identity of the species. The US program does this by trying to preserve the original gene pool, whereas the European program involves a form of mild selective breeding.

Mathematical geneticists in particular are very pessimistic about any system that augments the already high degree of human selection and the selection that occurs when the population limit at a breeding institution is reached. Selection may have the desired effect on one gene but has an unknown effect on the rest of the genome and may cause the loss of rare original genes.

This is indicated in the results of the founding and current population allele frequencies. Care has to be taken in interpreting the results, as the current allele frequencies do not come from an independent sample, however the general trend appears to be the loss of the rarer alleles in the current population, sometimes by quite a large proportion. This could mean the loss of these alleles with unknown consequences.

These facts seems to suggest that there is a high degree of variation in the allele frequencies of the founders which is probably due to a large amount of interbreeding between Przewalski and domestic horses on the Mongolian plains. It therefore seems futile to have management programs that are essentially trying to eradicate the domestic

genes out of the population. In trying to do this, unknown damage to the genome is taking place.

Chapter 3

A Gibbs EM Algorithm

Life is a gamble at terrible odds –
if it was a bet, you wouldn't take it.
(Tom Stoppard)

3.1 Introduction

In the previous chapter we introduced an EM algorithm and showed that it is an efficient way of finding the MLEs of the allele frequencies of the founding population. This allows us to make various inferences on interesting aspects of the population and its management. However, due to the computational complexity of the peeling algorithm, which was used in the estimation step, the procedure was carried out only on the two and three allele traits. In §1.14, we outlined the way in which MCMC methods can be used in pedigree analysis to simulate from the posterior distribution of interest. The method worked by using a MRF defined on the close relations in the pedigree of an individual.

We now propose to replace the peeling algorithm in the estimation step of the EM algorithm with the Gibbs Sampler. Guo and Thompson (1994) use a Gibbs EM algorithm to estimate parameters of a two allele mixed model which is one with added genetic effects and a vector of error effects. The authors note that the estimate of the A allele frequency is simply the expected proportion of A alleles in the founders.

Peeling provides the exact answer but the Gibbs Sampler provides an estimate which should converge to the exact answer and has the advantage of being readily generalised to more than three alleles when peeling becomes infeasible.

There are however problems when using a MCMC method in pedigree analysis.

Irreducibility of the MCMC Algorithm

The irreducibility criterion, which is needed to make the Markov Chain converge to the posterior, does not hold for multi-allelic traits. However, in §1.14.2 we gave a scheme for sampling from the wrong model by giving each of the zero transmission probabilities a

small positive probability, γ.

$$\tau_*(i_k, i_{f_k}, i_{m_k}) = \begin{cases} \tau(i_k, i_{f_k}, i_{m_k}) & \text{for } \tau > 0, \\ \gamma & \text{for } \tau = 0 \text{ and } \gamma > 0 \end{cases} \tag{3.1}$$

This model can produce inconsistencies – configurations that break the Mendelian laws of inheritance – but any such configurations are rejected. The resulting model is irreducible and has a stationary distribution proportional to the correct model. This leaves us to choose the parameter γ to control the rejection rate, which affects not only the run time of the simulation but also the serial correlation of the realisations.

Gilks et al. (1993) worries about producing inconsistencies too often to be of practical use in a large pedigree and this will be examined in §3.4 with the use of varying relaxation parameters.

Rejection Sampling

Gilks et al. (1993) also doubts the ability of rejection sampling to ensure visits to all irreducible subsets in the space of genotypic configurations on a large pedigree. We propose to check the Gibbs results with the MLEs that are available from the peeling process for the two and three allele traits. We investigate the accuracy of the simulated Gibbs results for different γ by comparing them with the accurate peeling algorithm results for the three allele traits before trying the Gibbs EM method on the higher allele traits for which we have no exact answers.

Multi-Modality

When examining the four and higher allele traits it is impossible to use the peeling algorithm to obtain the likelihood surface and check the results. In particular, if the surface is multi-modal, then we have to check that the EM process does not stick at a local maximum. We will use multiple starting points for π_1 and π_2 to give us greater confidence in the unimodal assumption of the likelihood surface.

Chain Mixing

As the dimension of the sample space increases then it may take astronomical sample sizes to get good estimates. We shall use a simple Gibbs Sampler and investigate the convergence of the algorithm using different numbers of Gibbs sweeps.

3.2 The Two Allele Traits

We implement the Gibbs EM algorithm on the two allele traits. We have shown in §1.14.2 that not all two allele traits have to be irreducible, so to guarantee convergence of the Markov Chain simulation we use a relaxation parameter. The only effect is to lengthen the

run time of the simulation due to rejecting any configuration inconsistent with Mendelian inheritance.

The procedure is to start with arbitrary allele frequencies and obtain an initial configuration consistent with observed data and the allele frequencies. We then run a Gibbs Sampler on the founders of the pedigree for a certain number of sweeps. After this has finished we have that number of genotype realisations for every founder. We then perform the maximisation procedure using a simple gene count on all the realisations to obtain the new estimates of the allele frequencies. This gives us our first EM iteration. We then repeat this procedure for however many iterations we deem necessary. In order for the sampler to get close to the stationary distribution from which we wish to sample and give good estimates, we should disregard a certain number of initial EM iterations.

For reasons outlined in §3.5 we use 10,000 Gibbs sweeps per EM iteration. We use a relaxation parameter of 0.005 for all of the two allele traits except for MPI and PEPD where a parameter of 0.0005 is used. This is because of an unacceptably high rejection rate giving rise to impractical run times and will be explained and investigated in §3.4. We adopt Geyer's suggestion of a 5% burn in period, so after a burn in period of 50 EM iterations, we perform 1000 EM iterations. The starting frequencies used to initialise the algorithm in each case are (0.5, 0.5).

Results

The estimates of the MLEs of the rarer founding allele frequencies are given in Table 3.1. The exact MLEs of the rarer founding allele frequencies which we obtained from the peeling technique are also shown for comparison.

Comparison of the MLEs of the Rarer Allele Using Exact and Simulated Results					
	Trait				
	ALB	BF	CA1	FUCa	GPI
Exact	0.1131	0.1066	0.1094	0.1341	0.1298
Gibbs Est	0.1132	0.1078	0.1086	0.1342	0.1305
Run Time (Secs)	48,566	59,012	67,078	104,599	40,542
	Trait				
	Hb	MPI	PEPB	PEPD	PGM1
Exact	0.3062	0.1977	0.3574	0.2264	0.1947
Gibbs Est	0.3059	0.1980	0.3567	0.2264	0.1927
Run Time (Secs)	43,717	35,979	43,178	58,375	41,664

Table 3.1: MLEs for the Rarer Allele of the Two Allele Traits: The table displays estimates of the MLEs of the rarest allele frequency derived from 1000 Gibbs EM iterations of 10,000 Gibbs sweeps per iteration. The starting frequencies used to initialise the algorithm in each case are (0.5, 0.5). The exact values for the MLEs of the rarer allele frequency obtained from the decimal search using the peeling algorithm in §2.2.1 are shown for reference in the top row. The run time is given in seconds.

Comparison Between Peeling and Gibbs

The MLEs obtained via the Gibbs Sampler do very well when compared to the exact numbers. Most of the estimates are correct to three decimal places.

Computational Time

In §2.2, we initially obtained the MLEs by finding the likelihood by running the peeling process for a number of points covering the sample space. To get the likelihood graphs in Figure 2-2 we used 20 points which took approximately five seconds per point. To get our predetermined accuracy of four decimal places we would have to run the peeling program 1000 times taking approximately one and a half hours.

We then introduced an EM algorithm which used 10 iterations to get a similar degree of accuracy and took approximately 20 minutes to obtain the exact MLEs given in Table 3.1.

Using our Gibbs EM algorithm, the estimates take approximately 12 hours which is considerably longer than the peeling results but the method allows us a great deal of flexibility. We can adjust several parameters – the number of Gibbs sweeps per EM iteration, the number of iterations and the relaxation parameter – all of which affect the efficiency of the algorithm, but most important of all is that the Gibbs EM algorithm can be easily generalised to higher dimensions.

3.3 The Three Allele Traits

In this section as well as using the Gibbs EM algorithm to obtain estimates of the MLEs for the three allele traits, we also investigate multi-modality and how the number of sweeps per iteration affects the accuracy of the Gibbs EM when compared to the peeling EM algorithm.

3.3.1 C71

We implement the Gibbs EM algorithm on the three allele trait C71. To get information on how accurate the Gibbs samples are, we only do a small number of EM iterations from multiple starting frequencies covering the sample space and compare the resulting estimates with the exact peeling answer for the same number of EM iterations from the same starting frequencies.

The Gibbs EM algorithm was run for 10 EM iterations. As we are dealing with frequencies, for any multi-allele trait they must sum to unity, so in practice, we have only two variable allele frequency parameters as the third frequency is fixed by the first two. To counter the danger of multi-modality, six different starting allele frequencies were used: (0.02,0.02), (0.02,0.49), (0.02,0.96), (0.49,0.49), (0.96,0.02) and (0.49,0.02). For reasons outlined in §3.4, all the simulations use a relaxation parameter of 0.005 which gives rise to a rejection rate of approximately 75%.

In order to assess the Gibbs Sampler six different numbers of Gibbs sweeps – 100, 250, 500, 1000, 5000 and 10,000 – were used per EM iteration. Generally the length of run time per iteration ranged from approximately five seconds for 100 sweeps, through to about one minute for 10,000 sweeps. However, getting the first iteration can take longer because the algorithm struggles to find consistent configurations when the initial frequencies are highly weighted in favour of one of the alleles, for example (0.96, 0.02, 0.02).

Results

Figure 3-1(a) gives the results for 100, 250 and 500 Gibbs sweeps per EM iteration and Figure 3-1(b) gives the results for 1000, 5000 and 10,000 Gibbs sweeps per EM iteration. The result after each iteration is denoted by a dot and the Gibbs EM iterations are joined by dashed lines. For comparison, we also show the exact results obtained by running 10 peeling EM iterations from each of the six starting frequencies where we use the peeling algorithm in the estimation step. The peeling EM iterations are joined with a solid line. The six different spurs for the starting locations are labelled 1 – 6 on Figure 3-1(a). The figures are bounded by the $x + y = 1$ line shown in the diagrams due to the unity condition of the allele frequencies. The grey lines are used for reference only.

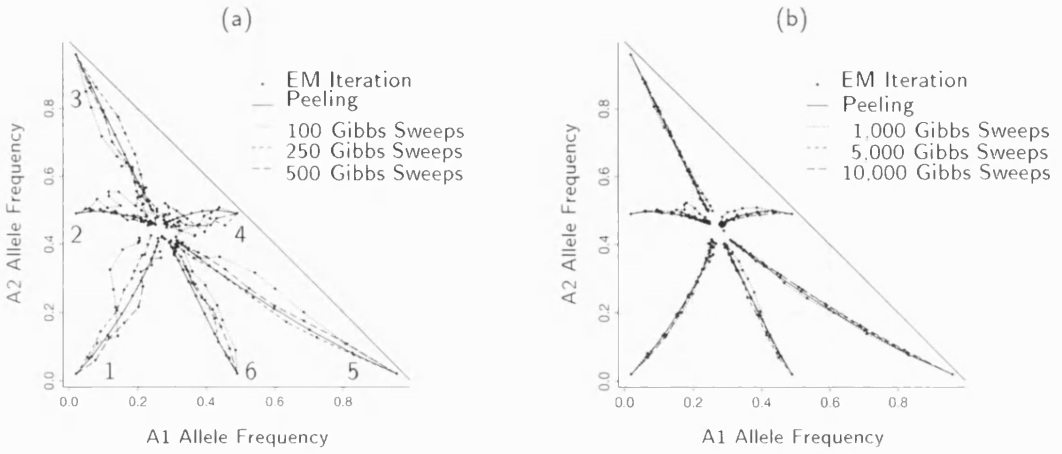


Figure 3-1: Convergence Plots for Trait C71: This figure compares 10 iterations of the exact peeling EM algorithm and estimates using various numbers of sweeps of the Gibbs Sampler per EM iteration. Six separate starting locations which are numbered 1 – 6 are used to pick up on any multi-modality of the likelihood surface. Convergence is evident. The grey lines are used for reference only.

In Figure 3-2 each numbered graph corresponds to the numbered spur of Figure 3-1. Care has to be used when interpreting the results as these graphs are not on the same scale.

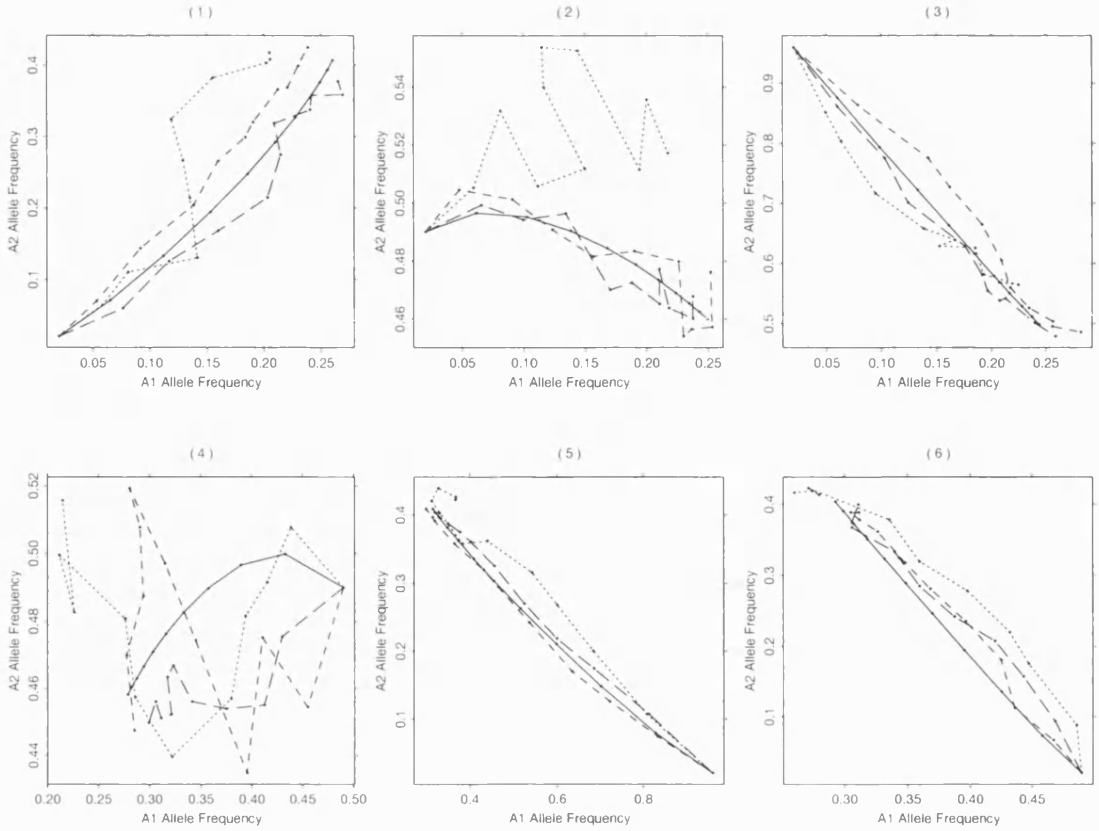


Figure 3-2: Close Up Convergence Plots: This figure shows six close up plots of each of the numbered spurs from Figure 3-1(a). Note that the plots are not on the same scale.

Comparison with Peeling

The simulated Gibbs EM iterations appear to perform very well. For the larger numbers of Gibbs sweeps per EM iteration shown in Figure 3-1(b), the lines for the peeling are indistinguishable from the lines for 5,000 and 10,000 Gibbs sweeps per EM iteration. Only when the number of sweeps per iteration is dropped to 1000 do the simulated lines begin to deviate from the exact. The trend is continued in Figure 3-1(a). Further reductions in the number of Gibbs sweeps make the lines more erratic and a good example is shown in Figure 3-2(1), where the line of 100 sweeps swings either side of the exact line. However, even 100 sweeps is enough tentatively to suggest that there is a single point of convergence.

Convergence

Since the simulated answers closely follow the exact answers, the Gibbs EM algorithm converges to the MLE (0.2687,0.4510) which we obtained from the peeling EM algorithm in Table 2.2. Even for small numbers of sweeps, the Gibbs EM algorithm does converge to

approximately the correct place, it is only when the process approaches the MLE that it does not make much further progress as the sampler is affected by the random simulations that are produced. This can be seen in Figures 3-2(2) and 3-2(4) which are the two graphs on a different scale due to the starting frequencies being close to the MLE. This suggests that for efficient computation purposes we can initially rattle through several EM iterations with small numbers of Gibbs sweeps per iteration and then increase the number of sweeps per iteration as we approach the MLE to obtain accuracy.

The convergence of the algorithm is rapid. To obtain the MLEs of the founding allele frequencies using the peeling EM method in §2.3.2, we used the current population estimates, which provided good initial estimates. After 10 EM iterations we were typically within 1–2% of the MLEs for trait C71. This time we are deliberately choosing allele frequencies that are far away from the MLEs to test the Gibbs EM algorithm and consequently, after 10 EM iterations we are not as near to the MLEs. However, we are typically within approximately 10% of the MLE after 10 iterations and by running for a few more iterations we would converge to the MLEs.

In this section we have used a loose definition of convergence. Strict convergence as far as MCMC algorithms are concerned could be obtained by the method if the number of MCMC sweeps increased in some way as we approached the MLE. Practically though we have found that 100,000 is enough.

Time Saving

In order to persuade the peeling EM to work for the three allele traits, we needed to break down the 13 founders into three groups and run all three batches to get one EM iteration. This regime takes approximately 10 minutes per iteration on a Sun Server 1000 and so a run of 10 iterations would take approximately 100 minutes. Running the Gibbs EM using a relaxation parameter of 0.005, 10 iterations of 1000 Gibbs sweeps takes approximately 40 minutes in total. So, unlike the two allele traits, we can now see that the Gibbs EM algorithm offers a computational saving over the peeling EM algorithm.

Conclusions

For the two allele traits we saw that the Gibbs Sampler achieves good estimates of the allele frequencies and this is shown again here. Having checked for multi-modality, we find that the simulated Gibbs EM estimates closely follow the exact values from the peeling algorithm and so will converge to the MLEs if the sampling is run for long enough and the number of sweeps is increased as we approach the MLEs. The Gibbs EM algorithm is computationally efficient and works even for extremely low numbers of Gibbs sweeps. This is due to the maximisation step which by gene counting effectively averages all the realisations from the Gibbs Sampler and so smoothes out the effect of unusual realisations that the sampling process might produce.

Consequently, the algorithm is rapid. To check for multi-modality only small numbers of sweeps are needed for a small number of EM iterations to suggest convergence to a single

point. Any increase in computer time would be better spent in increasing the number of initial frequencies to improve confidence in the unimodal assumption of the likelihood surface. To obtain accurate estimates of the MLEs we then have to increase the number of Gibbs sweeps per iteration after the process starts sampling from the marginal posterior of interest.

3.3.2 C72

The results for the Gibbs EM algorithm for trait C72 using the same regime as for the C71 trait in the previous section are shown in Figure 3-3. They show similar results to those for the C71 trait. The Gibbs EM simulations again follow the peeling EM exact answers, which indicates convergence to the MLE obtained from Table 2.2. The lines again become increasingly erratic as the number of sweeps drops but for 1000 sweeps per iteration we are still within approximately 10% of the MLE after only 10 EM iterations.

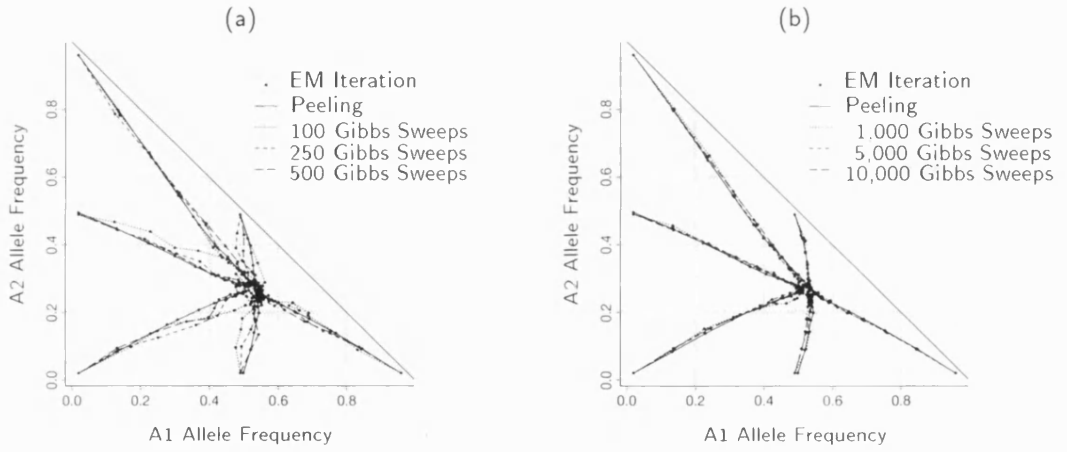


Figure 3-3: CONVERGENCE PLOTS FOR C72: This figure compares 10 iterations of the exact peeling EM algorithm and an estimate using various numbers of sweeps of the Gibbs Sampler per EM iteration. Six separate starting locations are used to pick up on any multi-modality of the likelihood surface. Convergence to the MLE is evident. The grey lines are used for reference only.

3.3.3 Pa

The results for the Gibbs EM algorithm for trait Pa are shown in Figure 3-4. We have used the same parameters as for the other three allele traits, however, for this trait, it takes approximately twice as long as the other two three allele traits to get each Gibbs EM iteration, because of a higher rejection rate. Again we have convergence to the MLE and the same characteristics as mentioned for the previous three allele traits.

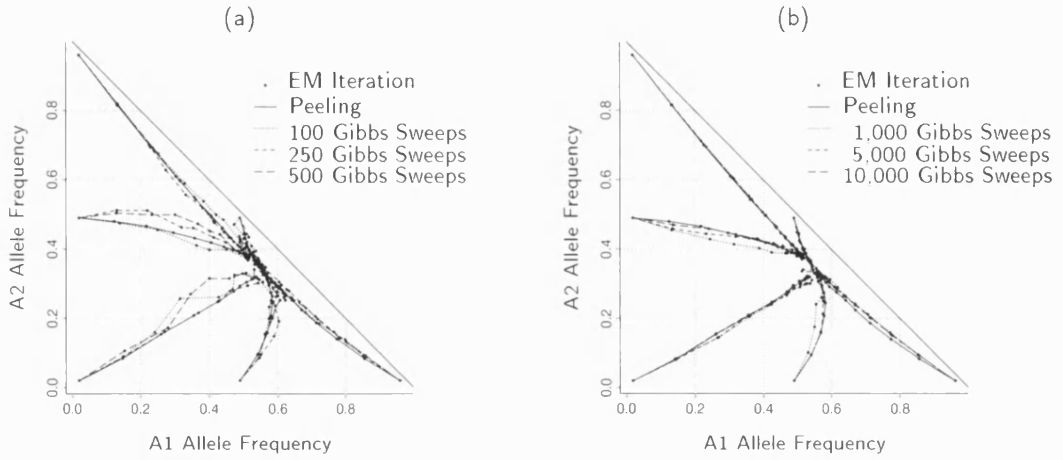


Figure 3-4: CONVERGENCE PLOTS FOR PA: This figure compares 10 iterations of the exact peeling EM algorithm and an estimate using various numbers of sweeps of the Gibbs Sampler per iteration. Six separate starting locations are used to pick up on any multi-modality of the likelihood surface. Convergence to the MLE is evident. The grey lines are used for reference only.

3.4 Relaxation Parameter

In this section we investigate the worries expressed by Gilks et al. (1993) about whether rejection sampling is of any practical use in a large pedigree. Although we do not have a very large pedigree, only 244 individuals, we will later consider a six allele trait which will put similar strains on the rejection sampling method. The effect of the relaxation parameter γ will be investigated as it is this parameter which controls the rejection rate and consequently the run time. We can use the exact answers of the peeling EM algorithm as a benchmark to see the effect of different γ.

3.4.1 Constant Number of EM Steps

The first stage is to investigate the effect of γ for a small number of Gibbs EM iterations. Low γ means that after we have found a feasible configuration we accept nearly every configuration that is generated. Higher γ means more configurations are rejected and so accepted configurations are less correlated.

Primarily in this section, we will be interested in the distance from the exact peeling EM estimate after 10 iterations for different γ. However, registering how far away we are from the MLE might give additional information on the number of Gibbs sweeps for efficient computational purposes. Further information about the number of Gibbs sweeps per EM iteration required can be obtained by running the algorithm with different numbers of sweeps per iteration.

The Gibbs EM algorithm was run on trait C71. 10 EM iterations with eight different γ parameters are used from initial allele frequencies (0.02, 0.02). Two numbers of Gibbs sweeps per iteration – 1000 and 10,000 – are used. The Cartesian distance is used to

measure the accuracy of the run of 10 EM iterations in relation to the MLE and the answer for the 10^{th} peeling iteration.

Results

The results obtained from the simulations using different γ are shown in Table 3.2. For each γ, the corresponding total number of sweeps, rejection rate, run time, finishing allele frequencies and the two Cartesian distances from the MLE (0.2687,0.4510) and 10^{th} peeling step (0.2601, 0.4063) are shown. The top half of the table gives the results using 1000 Gibbs sweeps per EM iteration and the bottom half gives the results for 10,000 Gibbs sweeps per EM iteration. The finishing allele frequency for allele 1 appears to be closer to its MLE of 0.2687 than the allele 2 estimate. This is because we started the process with initial frequencies of (0.02, 0.02) and the allele 2 frequency has further to go to reach its MLE.

Relaxation Parameter Effect

The results shown in the table indicate that as γ increases so does the rejection rate. Consequently, this increases the total number of Gibbs sweeps required as more configurations are generated that break the basic laws of inheritance and have to be rejected, increasing the run time.

Comparing similar rejection rates for both numbers of Gibbs sweeps – 1000 and 10,000 – we can see almost identical numbers. This is what we would expect as increasing the number of sweeps per iteration should have no effect on the rejection rate.

Comparing 1000 and 10,000 Gibbs Iterations per Step

A tenfold increase in the number of Gibbs sweeps means we would expect a similar increase in run time and this is shown in the table. The results for the total number of Gibbs sweeps and run time for the 10,000 sweeps per EM iteration are almost exactly ten times greater than the corresponding figures for the 1000 sweeps per EM iteration.

Comparing the values obtained for the Cartesian distance from the MLE for each run is inconclusive. Both sets of numbers are typically around 4×10^{-2}. and they show no trend as γ varies.

The 10^{th} peeling iteration represents where we should be after running the Gibbs EM algorithm under this regime and so we would expect the Cartesian distance to be smaller as we have not achieved convergence to the MLE with only 10 EM iterations. This is indicated in the table with typical values for both sets of figures of 1×10^{-2}. For the larger number of Gibbs sweeps we would expect MCMC to show greater convergence to the true answer as it will not be subject to a group of unusual simulations and this can be seen in the table. The distances from the 10,000 sweeps are slightly smaller but are a lot less variable than the corresponding values for 1000 sweeps.

Results from 10 EM Iterations Using 1000 Gibbs Sweeps Per EM Iteration						
γ ($\times 10^{-2}$)	Rej Rate (%)	Nos of Sweeps	CPU Time (sec)	Finishing Allele Freqs	Distance From MLE ($\times 10^{-2}$)	Distance From 10^{TH} Peeling It ($\times 10^{-2}$)
0.0005	0.10	10,010	46	0.2549, 0.4319	2.361	2.610
0.001	0.10	10,010	46	0.2414, 0.4096	4.959	1.899
0.005	1.17	10,118	46	0.2817, 0.3995	5.310	2.257
0.01	2.11	10,216	47	0.2590, 0.4081	4.397	0.215
0.05	13.73	11,591	56	0.2592, 0.4123	3.983	0.606
0.1	23.74	13,113	61	0.2649, 0.4040	4.715	0.532
0.5	75.36	40,588	194	0.2666, 0.3965	5.454	1.170
1.0	95.54	219,105	936	0.2570, 0.4147	3.817	0.892
Results from 10 EM Iterations Using 10,000 Gibbs Sweeps Per EM Iteration						
γ ($\times 10^{-2}$)	Rej Rate (%)	Nos of Sweeps	CPU Time (sec)	Finishing Allele Freqs	Distance From MLE ($\times 10^{-2}$)	Distance From 10^{TH} Peeling It ($\times 10^{-2}$)
0.0005	0.15	100,148	473	0.2557, 0.4151	3.822	0.982
0.001	0.21	100,205	473	0.2618, 0.4059	4.564	0.171
0.005	1.19	101,209	473	0.2681, 0.3991	5.193	1.076
0.01	2.58	102,650	487	0.2563, 0.4186	3.473	1.283
0.05	11.87	113,490	512	0.2663, 0.4078	4.328	0.632
0.1	22.79	129,514	588	0.2547, 0.4125	4.094	0.822
0.5	75.50	408,045	2002	0.2635, 0.4024	4.885	0.518
1.0	95.56	2,257,281	9603	0.2585, 0.4074	4.480	0.196

Table 3.2: Effect of the Relaxation Parameter: The table displays information about various relaxation parameters for trait C71. The top half of the table gives the results for 10 EM iterations using 1000 Gibbs sweeps per iteration and the lower half gives the results for 10 EM iterations using 10,000 Gibbs sweeps per iteration. In both cases initial allele frequencies (0.02, 0.02) are used. To provide a measure of accuracy the Cartesian distance is used as a measure of the accuracy of the finishing allele frequencies in relation to the MLE and from the 10^{th} peeling iteration.

When considering the accuracy by comparing the results to the 10^{th} peeling iteration answers, there appears to be a slight trend of increasing accuracy with increasing γ. This is reasonable as larger relaxation parameters generate simulations which have less serial correlation and consequently give more accurate answers.

3.4.2 Constant Number of Gibbs Iterations

Running the Gibbs EM process using a larger γ parameter incurs a longer run time. For this longer run time we could have achieved many more EM iterations using a lower γ parameter and consequently got a lot nearer to the MLE.

In this section we propose to run the Gibbs EM algorithm but instead of keeping the number of EM iterations constant at 10, we propose to keep the total number of Gibbs

sweeps approximately constant. Due to the rejection rate of inconsistent configurations, this means that for different γ parameters we achieve a different number of EM iterations. We will be interested in how close to the MLE we get.

The Gibbs EM algorithm is again run on the trait C71 with initial allele frequencies (0.02, 0.02) as before. We use 10,000 Gibbs sweeps per EM iteration, however, this time we have run the process with an additional four γ parameters to clarify the results. The total number of Gibbs sweeps was kept constant at approximately two million.

Results

The results obtained from running the above regime are given in Table 3.3. For each of the twelve different γ parameters, the corresponding values for the total number of EM iterations, the total number of Gibbs sweeps involved, the finishing allele frequencies at the end of the run and the Cartesian distance away from the MLE (0.2687,0.4510) are presented. For two million sweeps the CPU run-time is approximately three hours.

Results for a Fixed Computer Cost Using Different Relaxation Parameters				
γ ($\times 10^{-2}$)	Nos of EM Iterations	Total Nos of Sweeps	Finishing Allele Freqs	Distance From MLE($\times 10^{-3}$)
0.0005	200	2,001,828	0.2792, 0.4450	12.05
0.001	200	2,003,782	0.2734, 0.4450	7.65
0.005	195	1,968,440	0.2708, 0.4434	7.87
0.01	190	1,935,709	0.2738, 0.4477	6.04
0.05	180	1,976,312	0.2671, 0.4475	3.90
0.1	150	1,808,955	0.2671, 0.4539	3.30
0.5	75	2,020,710	0.2688, 0.4528	1.80
0.8	30	1,753,658	0.2705, 0.4533	2.86
0.9	20	2,083,180	0.2689, 0.4484	2.60
1.0	10	2,254,337	0.2585, 0.4074	44.80
1.1	6	2,185,883	0.2272, 0.3296	128.83
1.2	3	2,423,486	0.1524, 0.1930	283.01

Table 3.3: Relaxation Parameter Effect: The table displays information about various relaxation parameters for trait C71. This time the total number of Gibbs sweeps was kept approximately constant at two million and so the number of EM iterations varies. 10,000 Gibbs sweeps were used for each EM iteration from the starting allele frequencies (0.02, 0.02). The order of the Cartesian distance from the MLE is $\times 10^{-3}$.

Number of EM Iterations

The results shown in the table indicate that for a fixed number of Gibbs sweeps, increasing the relaxation parameter γ decreases the total number of EM iterations. For a very low γ of 0.0005×10^{-2} we achieve 200 iterations of 10,000 Gibbs sweeps and so throw away only 1828 configurations or 0.1%. For a high γ of 1.0×10^{-2} we have achieved only 10 EM iterations of 10,000 Gibbs sweeps and so have discarded 2,154,337 configurations which

corresponds to a rejection rate of 95.56%, identical to the result in the previous section.

For the two million iterations and a $\gamma < 1.0 \times 10^{-2}$ we have managed more than 10 EM iterations which we used in the previous section and would expect to be closer to the MLE. This is shown in the table with the Cartesian distances typically on the order of between 3–7 $\times 10^{-3}$.

Accuracy

For a fixed number of Gibbs sweeps the accuracy appears to be parabolic in γ. For small γ, the accuracy of the finishing allele frequencies is low with a Cartesian distance of 12.05 $\times 10^{-3}$. For increasing γ, the serial correlation drops and we see increasing accuracy. For example 0.5×10^{-3} gives a Cartesian distance of 1.8 $\times 10^{-3}$. However, as γ increases further, we start to decrease the number of EM iterations performed for the fixed two million samples because of the increasing rejection rate. This means that we do not perform a sufficient number of EM iterations to get close to the MLE. For example with a γ of 1.0×10^{-2}, we only achieve 10 EM iterations and the Cartesian distance is up to 44.80 $\times 10^{-3}$.

3.4.3 Conclusions

For a constant number of EM iterations, both 1000 and 10,000 Gibbs sweeps per iteration get us the same distance away from the MLE but the 10,000 provides greater accuracy in relation to the 10^{th} peeling estimates which is where we should be.

We would like to run the process with as high a γ as we can. Very small relaxation parameters gives high serial correlation and adversely affects accuracy. However, despite increased accuracy, larger γ means a longer run time. For a fixed computer cost this accuracy is approximately parabolic with respect to γ.

Balancing these two criterion means that in §3.3, we opted to run the Gibbs Sampler with a γ of 0.5×10^{-2}. This provides a reasonable computation time and approximately accepts every fourth pedigree configuration which balances the effect of serial correlation on accuracy.

3.5 MLEs of the Three Allele Traits

In this section we obtain MLEs for the founding allele frequencies for the three allele traits.

C71

Having checked for multi-modality in §3.3 and investigated the effect of the relaxation parameter in §3.4, we are now more confident of the accuracy of any estimates produced by the Gibbs EM algorithm. So in this section we run a longer Gibbs EM algorithm to obtain estimates of the MLEs of the founding allele frequencies and compare them with the accurate answers obtained via the peeling EM method.

To check accuracy further, we run the EM algorithm under two different schemes. We keep the total number of consistent sweeps fixed at ten million but one scheme uses 10,000 EM iterations with 1000 Gibbs sweeps per iteration and the other scheme uses 1000 EM iterations with 10,000 Gibbs sweeps per iteration. The initial frequencies for the algorithm are (0.333, 0.333, 0.333) and we use a 5% burn-in period which corresponds to 500 iterations in the first scheme and 50 iterations in the second scheme. A γ of 0.5×10^{-2} is used giving rise to a rejection rate of approximately 75%. The runs take approximately 33 hours of CPU time.

Results

For both schemes the estimates of the allele frequencies for an increasing number of Gibbs EM iterations taken after the burn-in period are shown in Table 3.4.

Results of the MLEs from the Gibbs EM Algorithm						
Nos	1000 Sweeps Per EM Iteration			10,000 Sweeps Per EM Iteration		
of Its	Allele 1	Allele 2	Allele 3	Allele 1	Allele 2	Allele 3
10	0.2712	0.4514	0.2774	0.2677	0.4504	0.2820
50	0.2692	0.4485	0.2823	0.2683	0.4511	0.2806
100	0.2709	0.4491	0.2801	0.2695	0.4506	0.2799
250	0.2688	0.4516	0.2796	0.2693	0.4509	0.2798
500	0.2698	0.4511	0.2791	0.2689	0.4519	0.2799
1000	0.2701	0.4512	0.2787	0.2687	0.4511	0.2802
5000	0.2691	0.4511	0.2798	-	-	-
7500	0.2690	0.4510	0.2800	-	-	-
10,000	0.2689	0.4509	0.2802	-	-	-
Exact	0.2687	0.4510	0.2803	0.2687	0.4510	0.2803

Table 3.4: Maximum Likelihood Estimates for Trait C71: The table displays estimates of the MLEs for the founding allele frequencies obtained from increasing numbers of Gibbs EM iterations for two different regimes; one regime uses 10,000 EM iterations of 1000 Gibbs sweeps per iteration and the other uses 1000 EM iterations of 10,000 Gibbs sweeps per iteration. For both systems, the starting allele frequencies for the EM algorithm were (0.333, 0.333, 0.333) but in order to get our sample close to the stationary distribution we have used a 5% burn in period. The exact MLEs obtained via the peeling algorithm are shown for comparison.

The convergence of the algorithm is rapid. Little change is seen in the estimates after 5,000 iterations under the first scheme and 500 for the second scheme. This suggest that we have achieved convergence.

Under both schemes we have reached accuracy to three decimal places after only 250 iterations but slightly greater accuracy is achieved using 10,000 sweeps per iteration.

C72

We now use the Gibbs EM algorithm to obtain MLEs for the C72 trait. In the previous section we found that a higher number of sweeps per iteration gives rise to slightly better

estimates and so we propose to run a similar regime as before but only using 10,000 sweeps per EM iteration.

For this trait similar regime using a γ of 0.005 would mean a run time of approximately one week. So we run 1000 EM iterations with 10,000 sweeps per iteration using a γ of 0.001. This gives rise to a rejection rate of approximately 60% and a run time of about a day.

Results

The estimates for an increasing number of Gibbs EM iterations after the burn-in of 50 iterations are shown in Table 3.5.

Results of the MLEs from the Gibbs EM Algorithm Using 10,000 Sweeps Per EM Iteration			
Nos of EM Its	Allele 1	Allele 2	Allele 3
10	0.5008	0.2838	0.2154
50	0.4985	0.2829	0.2186
100	0.4989	0.2824	0.2187
250	0.4984	0.2824	0.2192
500	0.4985	0.2825	0.2190
1000	0.4987	0.2825	0.2189
Exact	0.4988	0.2826	0.2188

Table 3.5: Maximum Likelihood Estimates for Trait C72: The table displays estimates of the MLEs for the founding allele frequencies for trait C72 obtained from increasing numbers of Gibbs EM iterations. Each EM iteration uses 10,000 Gibbs sweeps. The starting allele frequencies for the EM algorithm were (0.333, 0.333, 0.333) but in order to get our sample close to the stationary distribution we have used a 5% burn in and discarded the first 50 Gibbs EM iterations. The exact MLEs obtained via the peeling algorithm are shown for comparison.

The results are similar to those for trait C71. We converge extremely rapidly to the true allele frequencies. After only 50 EM iterations we are within 5×10^{-4} for all three estimates. There is also very little change in subsequent estimates which means that we have reached convergence. After 1000 iterations we are within 1×10^{-4} for all three estimates.

Pa

We use the same process for the trait Pa using only the regime which uses 1000 EM iterations with 10,000 sweeps per EM iteration. For this trait a γ of 0.005 gives a rejection rate of approximately 98% and run time of three days that is just barely acceptable.

Results

The estimates for an increasing number of Gibbs EM iterations after the burn-in of 50 iterations are shown in Table 3.6.

Results of the MLEs from the Gibbs EM Algorithm Using 10,000 Sweeps Per EM Iteration			
Nos of EM Its	Allele 1	Allele 2	Allele 3
10	0.5411	0.3564	0.1024
50	0.5419	0.3555	0.1026
100	0.5432	0.3544	0.1024
250	0.5433	0.3544	0.1023
500	0.5434	0.3544	0.1022
1000	0.5434	0.3544	0.1022
Exact	0.5435	0.3542	0.1023

Table 3.6: Maximum Likelihood Estimates for Trait Pa: The table displays estimates of the MLEs for the founding allele frequencies for the Pa trait obtained from increasing numbers of Gibbs EM iterations. Each EM iteration uses 10,000 Gibbs sweeps. The starting allele frequencies for the EM algorithm were (0.333, 0.333, 0.333) but in order to get our sample close to the stationary distribution we have used a 5% burn-in and discarded the first 50 Gibbs EM iterations. The exact MLEs obtained via the peeling algorithm are shown for comparison.

The results again show rapid convergence to the MLEs. Very little change is seen after 100 iterations which again gives us confidence in convergence of the Gibbs EM algorithm.

3.6 The Four Allele Trait

In the previous sections, with the aid of the correct answers, we demonstrated that the Gibbs Sampler could be used instead of the peeling algorithm in the estimation step of the EM algorithm to provide good estimates of the MLEs for the allele frequencies of the founding population of the Przewalski horse pedigree. In this section we proceed to investigate the four allele trait Serum Es. This time we do not have the luxury of comparing the results of the Gibbs EM estimates with those from peeling, as the peeling algorithm cannot cope with a four allele system for this pedigree. So we have no guarantee that any convergence will be to the MLE.

In this section we will firstly check for convergence of the algorithm using the method of multiple starting points. We aim for efficient computation and would like to use small runs with a small number of different sweeps per iteration which will provide confidence in the point estimates obtained from a long run of the Gibbs EM algorithm. However, we now now have an increase in dimensionality and so keep a relatively large number of sweeps per iteration as a safeguard against this.

3.6.1 Multi-Modality

In practice we only have three allele frequencies as parameters, as the fourth one is fixed by the first three and the unity condition. The Gibbs EM algorithm was run for 10 iterations from six initial allele frequencies: (0.97, 0.01, 0.01), (0.01, 0.97, 0.01), (0.01, 0.01, 0.97), (0.49, 0.49, 0.01), (0.49, 0.01, 0.49) and (0.01, 0.49, 0.49). We run the EM algorithm with

two different number of sweeps, 1000 and 10,000, per iteration which is probably larger than we need but balances our worries about the increase in dimensionality.

This time we found that using a relaxation parameter of either 0.001 or 0.005 as for the three allele traits produced too high rejection rates and consequently too long a run time to be computationally worthwhile. So we reduce the relaxation parameter to 0.0005 which gives rise to a rejection rate of approximately 85%. The run time for 10 EM iterations was approximately 15 minutes using 1000 Gibbs sweeps per iteration and two and a half hours using 10,000 Gibbs sweeps per iteration.

Results

As we increase the dimension of the sample space, we run into the problem of how best to display convergence of the Gibbs EM algorithm in 3D. A table of results obtained after the 10 EM iterations could be displayed but this would be rather messy and difficult to understand. So we use a triplot to display the results.

A triplot represents three parameters as the perpendicular distance from each edge of the equilateral triangle scaled so that they sum to one. For example, in Figure 3-5(a) we consider the beginning of the spur of the diagram which starts at the top vertex of the triplot. The allele 1 frequency is perpendicularly far away from the bottom edge of the triangle but is perpendicularly equally very close to the allele 2 and allele 3 frequency edges. This represents the starting frequency of approximately (0.98, 0.01, 0.01). From the triplot, we cannot read the values of the point of convergence as the three values have been scaled so that they sum to one. These plots are simply a good way of showing convergence of the Gibbs EM algorithm for a four allele trait. The fourth allele, which is not showed on the triplots also converged.

Figure 3-5 shows the results of the Gibbs EM algorithm for the above regime on two triplots. Figure 3-5 (a) represents 1000 Gibbs sweeps per EM iteration and Figure 3-5 (b) represents 10,000 Gibbs sweeps per EM iteration. The results of each EM iteration are joined by an arrowed line where the size of the arrow head is proportional to the size of the jump.

Convergence

From Figures 3-5 we can see that the six runs of 10 iterations of the Gibbs EM algorithm do show convergence. The 10,000 Gibbs sweeps tend to be a little smoother, which we would expect, and seem to show less variation at the point of convergence. With hindsight the worry about the increase in dimensionality warranting a larger number of Gibbs sweeps per EM iteration seems to be unfounded. Low numbers of sweeps per EM iteration, certainly 1000 and possibly even as low as 100, can still be used as a check for multi-modality.

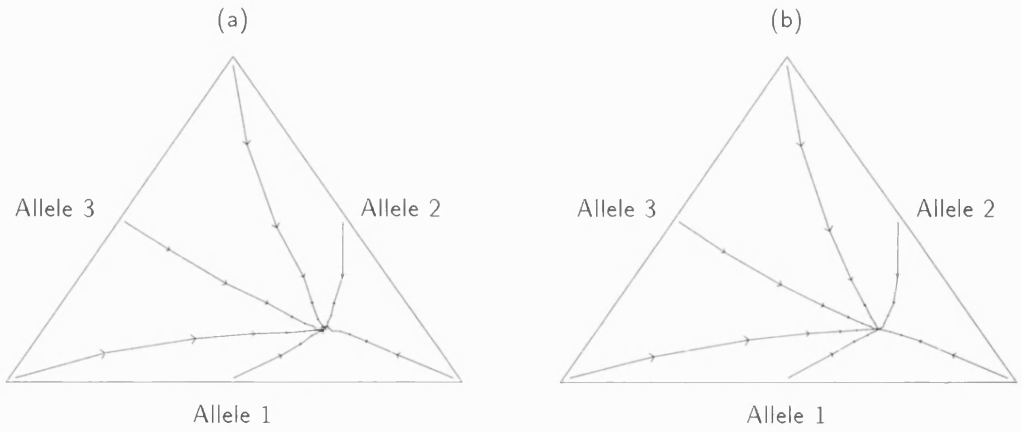


Figure 3-5: CONVERGE PLOTS FOR SERUM ES TRAIT: The figure shows 10 iterative steps of the Gibbs EM algorithm on the four allele trait Serum Es using multiple starting points. Plot (a) uses 1000 Gibbs sweeps per EM iteration and plot (b) uses 10,000 sweeps per EM iteration. The size of the arrow head is proportional to the size of the jump.

3.6.2 Maximum Likelihood Estimates

Now that we have checked for multi-modality, we run a long Gibbs EM simulation to get estimates of the MLEs of the founding allele frequencies.

For the previous traits we have run 1000 EM iterations of 10,000 Gibbs sweeps with a γ of 0.005. This provides a good estimate of the frequencies in a reasonable run time, but for this trait using that regime would take over a week. So using the short runs in the last section on multi-modality as a guide, a reduced scheme of 1000 EM iterations with 1000 sweeps per iteration and a γ of 0.0005 is used. This takes approximately twenty hours of CPU time to get estimates of the MLEs. The algorithm uses (0.25, 0.25, 0.25, 0.25) as its initial frequencies and discards the first 50 iterations.

Results

The MLEs of the allele frequencies for the above scheme are shown in Table 3.7 for an increasing number of EM iterations collected after the 50 EM iteration burn in period.

The convergence of the algorithm is still rapid. There is very little difference in the estimates between 50 and 1000 EM iterations and this must be due to the burn-in period getting us very close to the MLEs. Any slight changes in the estimates are then due to the random sampling of the MCMC algorithm.

3.7 The Five Allele Trait

We now consider the five allele trait Pr. With the increase in the dimension of the parameter space we have to be even more careful about the convergence of the Gibbs EM algorithm.

Results of the MLEs from the Gibbs EM Algorithm on Serum Es Using 1000 Sweeps Per EM Iteration				
Nos of Its	Allele 1	Allele 2	Allele 3	Allele 4
10	0.1460	0.2037	0.6019	0.0484
25	0.1464	0.2058	0.5992	0.0486
50	0.1466	0.2055	0.5994	0.0485
100	0.1465	0.2059	0.5992	0.0484
250	0.1466	0.2051	0.5999	0.0484
500	0.1466	0.2047	0.6004	0.0483
1000	0.1468	0.2044	0.6006	0.0483

Table 3.7: Maximum Likelihood Estimates for Trait Serum Es: The table displays the MLEs for the founding allele frequencies for trait Serum Es obtained using an increasing number of Gibbs EM iterations. Each EM iteration uses 1000 Gibbs sweeps. The starting allele frequencies were (0.25, 0.25, 0.25, 0.25) but in order to get our distribution close to the stationary distribution we have used a 5% burn-in and discarded the first 50 EM iterations.

3.7.1 Multi-Modality

Once again we have one less parameter to contend with than the dimensionality of the sample space due to the unity condition that the allele frequencies must show.

The Gibbs EM algorithm was run for 10 iterations from numerous starting points; these consisted of all the combinations where one of the allele frequencies was 0.96 and the remaining four were 0.01, and where two of the allele frequencies were 0.485 and the remaining three were 0.01. We ran the EM algorithm again with two different number of Gibbs sweeps per iteration to give an idea of how long a run to find the point estimates would take. A γ of 0.0005 gave a reasonable run time of four minutes for 10 iterations of 1000 sweeps and approximately 40 minutes for 10 iterations of 10,000 sweeps. We note that this is shorter than for the four allele trait with similar run parameters. As previously, a large proportion of the run time was used in getting the first EM iteration when the allele frequency is highly weighted in favour of one of the alleles. Again we use large number of sweeps per iteration as a safeguard against the increase in dimensionality.

Results

In order to display 5D on paper we use a convergence plot matrix. This plots every combination of pairs of alleles. Each panel in the matrix is a plot of one variable against another; for example the bottom left plot of Figure 3-6 plots allele 1 on the x axis and allele 5 on the y axis, the plot is duplicated in the top right corner but with the axes swapped. Each panel is bounded by the line $x + y = 1$ and again the grey lines on each of the plots are for reference only. Some spurs will have more than one pair of 10 Gibbs EM iterations because of the repetition of some of the allele frequencies in order to guarantee that other graphs obtain a new spur.

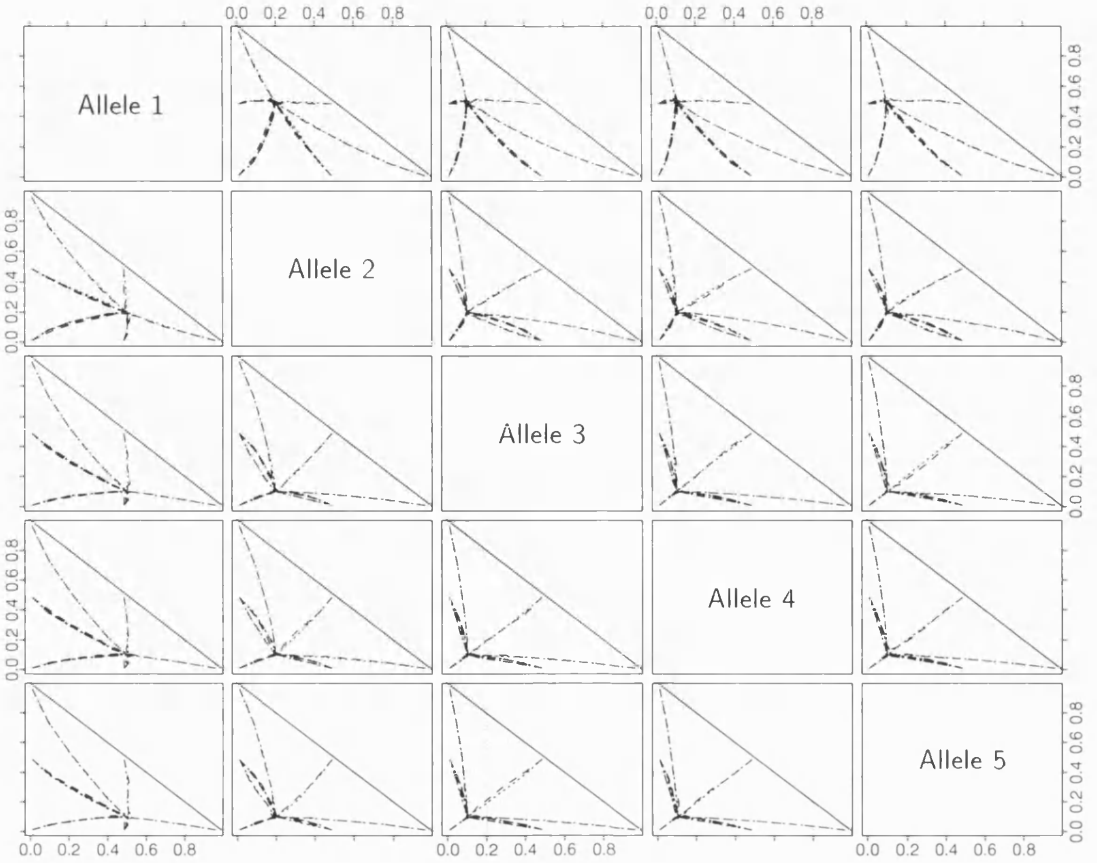


Figure 3-6: CONVERGE PLOTS FOR TRAIT PR: The figure shows allele by allele plots of the iterative steps using the Gibbs EM algorithm on the five allele trait Pr from multiple starting points. The smaller hatched lines use 1000 Gibbs sweeps per iteration and the larger hatched lines use 10,000 Gibbs sweeps per iteration.

Convergence

From Figure 3-6 we see that the multiple runs of 10 Gibbs EM iterations from different starting frequencies do show convergence. There appears to be very little difference between the 1000 and 10,000 lines with the largest difference shown in one of the spurs in the allele 2 against allele 3 plot.

3.7.2 Maximum Likelihood Estimates

Now we have shown that there is a reduced risk of multi-modality and have confidence in convergence of the Gibbs Sampler, we run a long simulation to obtain accurate estimates of the MLEs of the founding allele frequencies.

We start the Gibbs EM algorithm with initial frequencies of 0.2 for each allele. Discarding the first 50 EM iterations we collect the next 1000 and use these to obtain the estimates. We run both 1000 and 10,000 Gibbs sweeps per iteration for comparison and

use a relaxation parameter of 0.0005 which gives rise to a rejection rate of approximately 33%. The run time for 1000 EM iterations using 1000 sweeps per iteration is four hours and for 10,000 sweeps per iteration is approximately 40 hours.

Results

The results of the above regimes are shown in Table 3.8. The table shows the results of the estimates of the MLEs for the founding allele frequencies for increasing numbers of Gibbs EM iterations taken after the initial burn in period of 50 iterations.

Results of the MLEs from the Gibbs EM Algorithm On Pr Using 1000 Sweeps Per EM Iteration					
Nos of Its	Allele 1	Allele 2	Allele 3	Allele 4	Allele 5
10	0.5025	0.1932	0.1037	0.0969	0.1037
50	0.5043	0.1960	0.1000	0.0992	0.1004
100	0.5059	0.1939	0.1002	0.0997	0.1003
250	0.5075	0.1943	0.0996	0.0989	0.0997
500	0.5070	0.1937	0.0997	0.0998	0.0998
1000	0.5070	0.1935	0.0999	0.1000	0.0996
Results of the MLEs from the Gibbs EM Algorithm On Pr Using 10,000 Sweeps Per EM Iteration					
Nos of Its	Allele 1	Allele 2	Allele 3	Allele 4	Allele 5
10	0.5032	0.1970	0.0993	0.0993	0.1013
50	0.5052	0.1945	0.1004	0.1002	0.0998
100	0.5058	0.1942	0.1001	0.1003	0.0997
250	0.5057	0.1943	0.1005	0.0999	0.0996
500	0.5055	0.1944	0.1005	0.1000	0.0995
1000	0.5054	0.1944	0.1006	0.1001	0.0995

Table 3.8: Maximum Likelihood Estimates for Trait Pr: The table displays estimates of the MLEs for the founding allele frequencies for trait Pr obtained using an increasing number of Gibbs EM iterations. Two regimes of the algorithm were used; the top half of the table uses 1000 Gibbs sweeps per EM iteration and the bottom half of the table uses 10,000 Gibbs sweeps per EM iteration. In each case the starting frequencies were (0.2, 0.2, 0.2, 0.2, 0.2) but in order to get close to the stationary distribution we have used a 5% burn-in and discarded the first 50 EM iterations.

Again we seem to have reached a point of convergence as little change is noted in the estimates after 500 and 1000 EM iterations. If we compare the results for both regimes after 1000 iterations we can see that there are small differences; for example, allele 1 shows a difference of 0.0016. We have more confidence in the estimates obtained from the 10,000 sweeps per EM iteration regime, since the MCMC has been run for longer.

3.8 The Six Allele Trait

We now perform the Gibbs EM algorithm on the six allele trait Tf.

3.8.1 Multi-Modality

We again run the Gibbs EM algorithm for 10 EM iterations from numerous starting frequencies to check for multi-modality and to provide us with the parameters needed for efficient computation of the MLEs from the long run of the Gibbs EM algorithm. The starting frequencies consisted of all the combinations where one of the allele frequencies was 0.95 and the remaining five were 0.01, and where two of the allele frequencies were 0.48 and the remaining four were 0.01.

In the previous sections we have worried that an increase in dimensionality could affect the number of sweeps that we have to use in these checks for multi-modality. To test this we run the EM algorithm with three different numbers of sweeps per iteration: 100, 1000 and 10,000. A γ of 0.0005 gave a reasonable rejection rate of approximately 41% and gave rise to a run time of two minutes for 10 iterations of 100 sweeps, 20 minutes for 10 iterations of 1000 sweeps and approximately three hours for 10 iterations of 10,000 sweeps. Again a large proportion of the run time was used in getting the first EM iteration when the allele frequency is highly weighted in favour of one of the alleles.

Results

We again use a convergence plot matrix to display the convergence in 6D on paper. Each panel is again bounded by the line $x + y = 1$ and again the grey lines on each of the plots are for reference only.

The results of 100 and 1000 sweeps per iteration are shown in Figure 3-7 and the results of 1000 and 10,000 are shown in Figure 3-8.

The plots for 100 sweeps become erratic as we approach the point of convergence but still show convergence. Greater accuracy is observed with higher numbers of sweeps per iteration and the estimates get closer to the point of convergence before starting to vary. However, as this is just a check for multi-modality and tuning of the main run parameters, for efficient computation we can still get away with a very low number of Gibbs sweeps per EM iteration even for this six allele trait.

3.8.2 Maximum Likelihood Estimates

Now we have shown that there is a reduced risk of multi-modality and have greater confidence in convergence of the Gibbs EM algorithm to the MLEs, we run a long simulation to obtain good estimates of the founding allele frequencies.

We start the Gibbs EM algorithm with initial frequencies of 0.166 for each allele. Discarding the first 50 EM iterations we collect the next 1000 and use these to obtain the estimates. We run both 1000 and 10,000 Gibbs sweeps per iteration for comparison and use a relaxation parameter of 0.0005 which gives rise to a rejection rate of approximately 40%. The run time for 1000 iterations of 1000 sweeps was approximately seven hours and 1000 iterations of 10,000 sweeps was approximately three days.

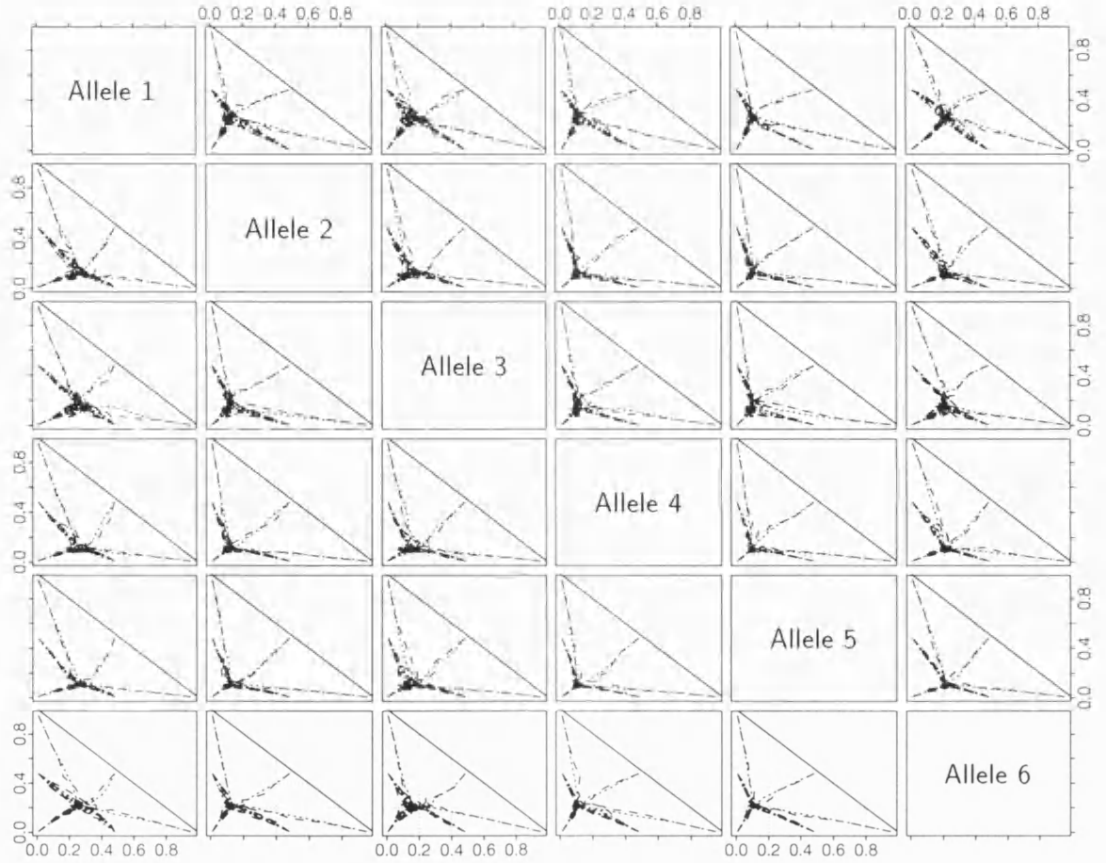


Figure 3-7: CONVERGE PLOTS FOR TRAIT TF: The figure shows allele by allele plots of 10 iterative steps of the Gibbs EM algorithm on the six allele trait Tf from multiple starting points. Smaller hatched lines use 100 Gibbs sweeps per iteration and the larger hatched lines use 1000 Gibbs sweeps per iteration.

Results

The results of the above regimes are shown in Table 3.9. The table shows the results of the allele frequencies for increasing numbers of EM iterations taken after the initial burn-in period of 50 EM iterations. For four of the alleles for this trait the differences in the estimates for 500 and 1000 iterations are negligible. However, allele 1 and especially allele 3, which shows a difference of 0.0015, for 10,000 sweeps from 500 to 1000 iterations, may not have yet converged to four decimal places. If further accuracy is needed a longer run is suggested. Again, we see that there are differences in the estimates obtained after 1000 EM iterations for the two regimes but they remain very small with the largest being 0.0008 for allele 6. We have more confidence in the estimates from the 10,000 sweeps per iteration but as the differences are so small, they would make little difference in practice.

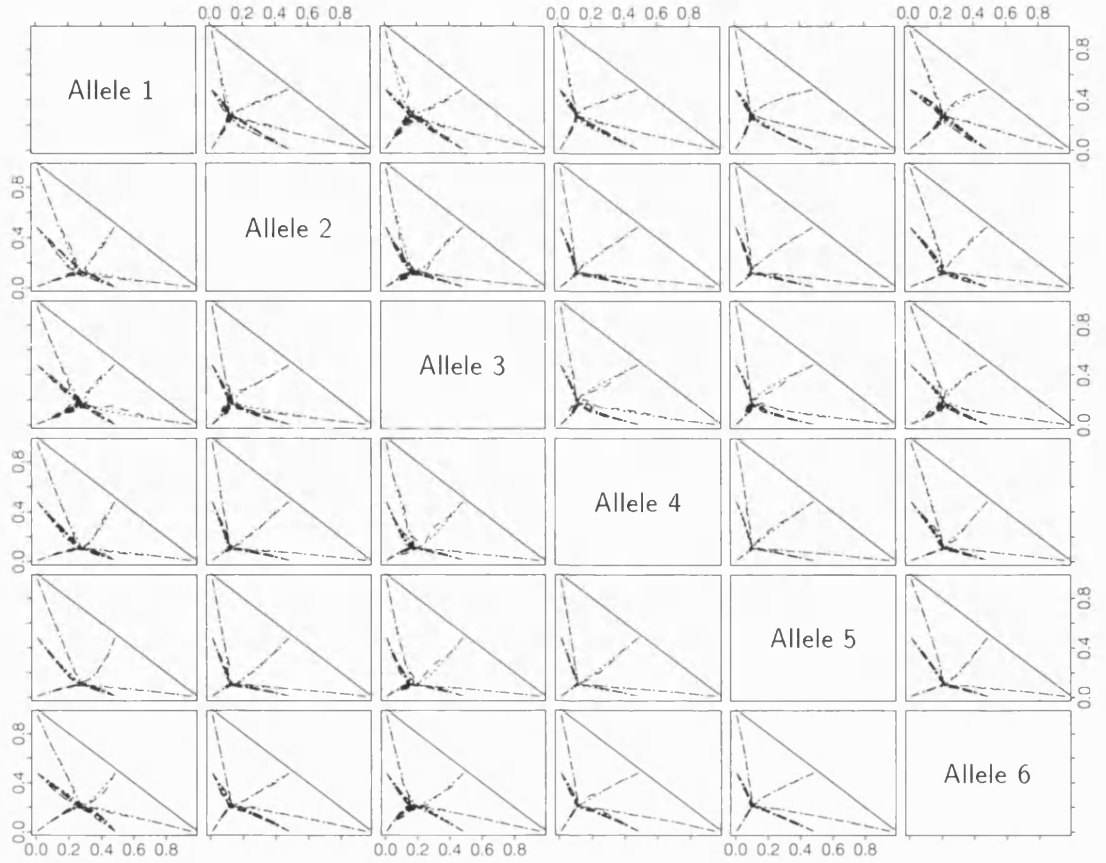


Figure 3-8: Converge Plots for Trait Tf: The figure shows allele by allele plots of 10 iterative steps of the Gibbs EM algorithm on the six allele trait Tf from multiple starting points. The smaller hatched lines use 1000 Gibbs sweeps per iteration and the larger hatched lines use 10,000 Gibbs sweeps per iteration.

3.9 Biological Implications

In this chapter, we have extensively investigated the use of the Gibbs Sampler and its ability to obtain the MLEs of the founding allele frequencies of the PH pedigree. However, the investigation was instigated by a simple biological question about management programs so it is interesting to ask whether by looking at the six founding groups of the population and the differences between the founding and current population frequencies we can say anything about the validity of two management programs.

3.9.1 Founding Group Estimates

In §2.3.1 and §2.3.2 we used the peeling process to obtain the allele frequencies for the six founding groups for all the two and three allele traits. This was done by running a further step of the EM algorithm from the best founding allele frequency estimates and then for each group doing a gene count on the individuals in that group. The method

Results of the MLEs Using the Gibbs EM Algorithm On Tf Using 1000 Sweeps Per EM Iteration						
Nos of Its	Allele 1	Allele 2	Allele 3	Allele 4	Allele 5	Allele 6
10	0.2689	0.1146	0.1992	0.1095	0.1034	0.2045
50	0.2689	0.1226	0.1844	0.1103	0.1012	0.2125
100	0.2736	0.1230	0.1770	0.1100	0.1021	0.2142
250	0.2759	0.1239	0.1759	0.1088	0.1020	0.2135
500	0.2747	0.1229	0.1800	0.1085	0.1016	0.2124
1000	0.2755	0.1227	0.1808	0.1084	0.1014	0.2113
Results of the MLEs Using the Gibbs EM Algorithm On Tf Using 10,000 Sweeps Per EM Iteration						
Nos of Its	Allele 1	Allele 2	Allele 3	Allele 4	Allele 5	Allele 6
10	0.2710	0.1167	0.1989	0.1074	0.1004	0.2058
50	0.2750	0.1201	0.1878	0.1077	0.1000	0.2095
100	0.2749	0.1221	0.1844	0.1078	0.1005	0.2103
250	0.2746	0.1221	0.1834	0.1079	0.1009	0.2111
500	0.2753	0.1222	0.1818	0.1079	0.1010	0.2118
1000	0.2762	0.1222	0.1803	0.1082	0.1011	0.2121

Table 3.9: Maximum Likelihood Estimates for Trait Tf: The table displays estimates of the MLEs for the founding allele frequencies for trait Tf obtained using an increasing number of Gibbs EM iterations. Two regimes of the algorithm were used; the top half of the table uses 1000 Gibbs sweeps per EM iteration and the bottom half of the table uses 10,000 Gibbs sweeps per EM iteration. The starting frequencies were (0.166, 0.166, 0.166, 0.166, 0.166) but in order to get close to the stationary distribution we have used a 5% burn-in and discarded the first 50 EM iterations.

breaks down for higher allele traits because of the huge amounts of memory used by the peeling routine.

Having shown that the Gibbs Sampler can obtain very good estimates of the founding allele frequencies, we can use it to obtain estimates of the allele frequencies for the six founding groups for the four, five and six allele traits. The method is similar to the one used in §2.3.1 and §2.3.2. Using the best estimate of the founding allele frequencies from the Gibbs EM algorithm, we can then run another 10,000 consistent Gibbs sweeps – in effect another Gibbs EM iteration – and perform a within group gene count on these realisations to get the group allele frequency estimates.

SerumEs

In this section we perform the above regime on the four allele trait SerumEs. From Table 3.7 we obtain our best estimates of the founding allele frequencies – the estimates corresponding to 1000 iterations – and use these to obtain an initial configuration from which to run the Gibbs Sampler. We have previously found in the EM structure for this trait that 10,000 consistent Gibbs sweeps per EM iteration with a γ of 0.005 gives good results and propose to base our founding group estimates on this regime. This gives a run time of approximately five minutes. Despite being very close to the MLEs, and hence the

distribution that we wish to sample from, we still perform a burn-in period of 5% in order to make the chain forget this initial configuration and so disregard 500 realisations before taking the 10,000. After taking these 10,000 consistent realisations on the 13 founders, for each group we perform a within group gene count on the individuals in that group to obtain the estimates. These are given in Table 3.10.

Allele Frequency Estimates of the Six Founding Groups Obtained Using the Estimates from the Gibbs EM Algorithm								
			Founding Group					
Trait	n	Allele	1	2a	2b	3	4	5
SerumEs	142	1	0.0896	0.2418	0.2156	0.0788	0.2643	0.0756
		2	0.2519	0.2178	0.1479	0.2579	0.2396	0.1052
		3	0.4669	0.5091	0.6144	0.6390	0.4614	**0.7945**
		4	**0.1917**	0.0313	0.0221	0.0243	0.0347	0.0248

Table 3.10: Allele Frequency Estimates for the Six Founding Groups: The allele frequency estimates of the six founding groups in the pedigree indicated in Figure 2-1 for trait SerumEs are shown. They are obtained by running 10,000 consistent Gibbs sweeps from the estimates of the MLEs of the founding allele frequencies from the 1000 iterations row from Table 3.7 and then performing a within group gene count to obtain the estimates. The column n represents the number of individuals for which we have phenotypic information. Results that differ by approximately 0.1 or more are shown in bold.

Results

The results for the SerumEs trait are similar to those from the two and three allele traits in that there is huge variation in the founding group allele frequencies. In fact, there is more variation in this trait than in any of the previously examined traits; for example, allele 3 ranges between approximately 0.45 and 0.80. In the description of the founding groups for the two and three allele traits, we highlighted in bold any results that differed from the others by approximately 0.1 or more to indicate unusual results. Here, because the range is so large only a few outliers are picked out; for example only group 5 in allele 3 and group 1 in allele 4 are highlighted. These groups do contain one of the three designated interesting founders – group 1 contains the domestic horse and group 5 contains the horse brought into the population later and from a different location – but the results exhibit so much variation that it is difficult to draw inference about these horses. However, this large variation reinforces inferences made in §2.3 about the huge variation in the founding population probably due to large amounts of interbreeding between domestic and Przewalski horse on the Mongolian plains. This again suggests that the management programs, which are based on so-called 'pure' and 'contaminated' horses and both involve an additional mild form of selection, are not justified.

Pr

Using the method outlined above we obtain estimates of the allele frequencies for the six founding groups for the five allele trait Pr. We again use the MLEs of the founding allele population obtained from 1000 iterations of the Gibbs EM algorithm shown in Table 3.8, perform a further 10,000 consistent Gibbs sweeps and then gene count within groups. A γ of 0.005 and a burn-in of 5% is used which gives rise to a run time of approximately five minutes. The results are shown in Table 3.11.

Allele Frequency Estimates of the Six Founding Groups Obtained Using the Estimates from the Gibbs EM Algorithm								
			Founding Group					
Trait	n	Allele	1	2a	2b	3	4	5
Pr	140	1	0.4308	0.5812	0.7040	0.6659	0.3495	0.3098
		2	0.1106	0.1645	0.1068	0.1409	0.2838	**0.5409**
		3	0.1847	0.1052	0.0792	0.0624	0.0692	0.0507
		4	**0.2200**	0.0766	0.0614	0.0584	0.0700	0.0504
		5	0.0539	0.0725	0.0487	0.0724	**0.2276**	0.0483

Table 3.11: Allele Frequency Estimates for the Six Founding Groups: The allele frequency estimates of the six founding groups in the pedigree indicated in Figure 2-1 for trait Pr are shown. They are obtained by running 10,000 consistent Gibbs sweeps from the estimates of the MLEs of the founding allele frequencies from the 1000 iteration row of Table 3.8 and then performing a within group gene count to obtain the estimates. The column n represents the number of individuals for which we have phenotypic information. Results that differ by approximately 0.1 or more are shown in bold.

Results

The results for the Pr trait are similar to those described above for trait SerumEs with huge variation between the groups. For example, allele 1 has a range of approximately 0.30 to 0.70. This again makes our *ad-hoc* bolding of unusual results rather difficult but three appear to stand out - group 1 for allele 4, group 4 for allele 5 and group 5 for allele 2. These groups are the three groups that contain one of the interesting horses but the large variation between all of the groups makes it difficult to draw any inference.

Tf

We continue our method to find estimates of the founding group allele frequencies for the six allele trait Tf. We again use the estimates of the founding allele frequencies obtained from 1000 iterations of the Gibbs EM algorithm shown in Table 3.9 from which to run the 10,000 consistent Gibbs sweeps. A γ of 0.005 and a burn-in of 5% gives rise to a run time of approximately five minutes. The results for the six founding groups are shown in Table 3.12.

Allele Frequency Estimates of the Six Founding Groups Obtained Using the Estimates from the Gibbs EM Algorithm								
			Founding Group					
Trait	n	Allele	1	2a	2b	3	4	5
Tf	140	1	0.3081	0.2905	0.1993	0.3672	0.2515	0.1665
		2	0.0752	0.0912	0.0790	**0.3133**	0.1209	0.0591
		3	0.1943	0.2860	0.2273	0.1111	0.1399	0.0895
		4	**0.2319**	0.0877	0.0713	0.0553	0.0819	0.0573
		5	0.0613	0.0828	0.0550	0.0510	0.0727	**0.5186**
		6	0.1293	0.1618	0.3682	0.1022	0.3331	0.1091

Table 3.12: Allele Frequency Estimates for the Six Founding Groups: The allele frequency estimates of the six founding groups in the pedigree indicated in Figure 2-1 for trait Tf are shown. They are obtained by running 10,000 consistent Gibbs sweeps iteration from the estimates of the MLEs of the founding allele frequencies from the 1000 iterations of the Gibbs EM shown in Table 3.9 and then performing a within group gene count to obtain the estimates. The column n represents the number of individuals for which we have phenotypic information. Results that differ by approximately 0.1 are shown in bold.

Results

The results for the Tf trait are similar to what we have seen before. There is a large variation between group estimates indicative of large variety in the ancestral population on the Mongolian plains. Three results have been highlighted and occur in groups 1, 3 and 5. Two of these again include one of the three interesting horses but group 3 is meant to be 'pure' Przewalski horse. This reinforces the inference made in the SerumEs section about the lack of justification for the current management programs.

3.9.2 Founding and Current Populations

In §2.1.2 we outlined four things that should be borne in mind when trying to maximise the genetic diversity of a small captive population – founder effect, genetic drift, inbreeding and selection. They usually work in conjunction to reduce the genetic variability of captive populations and can cause the loss of rare alleles.

In §2.3 we investigated two different measures of the population allele frequencies – the founding and current population. By comparing these two measures we can see the way that the allele frequencies have drifted through time. We can do a similar thing here for these multi-allele traits. The estimates of the MLEs of the founding allele frequencies obtained from the Gibbs EM algorithm can be compared with the current population estimates obtained from a simple gene count on the individuals in the pedigree for which phenotypic information is available.

SerumEs

The results for the four allele trait SerumEs for the two methods of obtaining the population allele frequencies are given in Table 3.13.

The Population Allele Frequency Estimates				
Population	Allele			
	1	2	3	4
Founding	0.1468	0.2044	0.6006	0.0483
Current	0.1796	0.2078	0.5810	0.0317

Table 3.13: Population Allele Frequency Estimates for SerumEs: The table displays two different estimates of the population allele frequencies. The founding population estimates are obtained from the Gibbs EM algorithm in Table 3.7. The current population estimates are obtained from a gene count on the individuals for which phenotypic information is available.

Results

We can see that the two estimates are quite similar with the largest difference of approximately 0.03 occurring for allele 1. It is interesting to note that the rarest allele, allele 4, has become slightly rarer with a drop of approximately 0.016.

Pr

The results for the five allele trait Pr for the two methods of obtaining the population allele frequencies are given in Table 3.14.

The Population Allele Frequency Estimates					
Population	Allele				
	1	2	3	4	5
Founding	0.5054	0.1944	0.1006	0.1001	0.0995
Current	0.7250	0.1107	0.0750	0.0679	0.0214

Table 3.14: Population Allele Frequency Estimates for Pr: The table displays two different estimates of the population allele frequencies. The founding population estimates are obtained from the Gibbs EM algorithm in Table 3.8. The current population estimates are obtained from a gene count on the individuals for which phenotypic information is available.

Results

This trait shows large differences between the founding and current population and is a good example of why for ancestral inference on the founders it is vital to find the founding MLEs, as using the current estimates could give misleading inference. The most common allele, allele 1, shows a rise of approximately 0.22 at the expense of all the other alleles which show sizeable decreases. This is important as alleles 3, 4 and 5 were quite rare in the founding population and in particular allele 5 is now in danger of disappearing altogether in the current population.

Tf

The results for the six allele trait Tf for the two methods of obtaining the population allele frequencies are given in Table 3.15.

The Population Allele Frequency Estimates						
Population	Allele					
	1	2	3	4	5	6
Founding	0.2762	0.1222	0.1803	0.1082	0.1011	0.2121
Current	0.2929	0.2428	0.3071	0.0286	0.0107	0.1179

Table 3.15: Population Allele Frequency Estimates for Tf: The table displays two different estimates of the population allele frequencies. The founding population estimates are obtained from the Gibbs EM algorithm in Table 3.9. The current population estimates are obtained from a gene count on the individuals for which phenotypic information is available.

Results

For this trait, there are again some substantial differences between the founding and current population estimates. Alleles 2 and 3 show increases in frequency of approximately 0.12 but this is at the expense of alleles 4, 5 and 6 which all show a reasonable decrease. Most importantly is that alleles 4 and 5 frequencies have become very low and are in real danger of disappearing from the population.

3.10 Conclusions

In this chapter we have shown that the Gibbs Sampler can be used to estimate the MLEs of the founders' allele frequencies of a pedigree. We have developed a method using the Gibbs Sampler nested in a simple EM algorithm. The method samples from the marginal distribution of the founders' genotypes and estimate the MLEs of the founding allele frequencies.

Using a Gibbs Sampler in the algorithm has advantages in that, unlike peeling, they can be generalised to multi-allele traits using the relaxation method. However, in order to get confidence in the estimates it produces, checking has to be done along the way.

Throughout the sampling in this chapter we have used four decimal places as a benchmark against which to run the algorithms. For many purposes this may be considered overkill and less accuracy might be needed which would reduce the run time of the algorithms and make them more efficient.

Gibbs EM Algorithm

For the Gibbs EM algorithm, multi-modality of the likelihood surface must be checked for and we have found that multiple starting points around the sample space is a good and computationally efficient way of doing this. As we are only interested if convergence appears to occur, even in high dimensions very low numbers of Gibbs sweeps can be used

with a low number of iterations. For all the alleles it appears that 10 EM iterations of 100 consistent Gibbs sweeps per iteration is sufficient provided that lots of starting points are used. Consequently, any increase in computing time would be better spent choosing more initial frequencies around the space to give more confidence in the unimodality assumption rather than in increasing the number of consistent Gibbs sweeps per EM iteration.

Doing these short runs also allows us to check the run parameters that we are going to use in the long run of the Gibbs EM algorithm, which provides the estimates of the MLEs of the founding allele frequencies. The most important of these is the relaxation parameter γ. It controls the rejection rate, which has the most effect on the run time of the EM algorithm. We have shown that we would like to run with as high a parameter as possible to reduce the serial correlation and improve accuracy of the estimates but this involves prohibitively long run times. So we come to a compromise and run the Gibbs EM algorithm with as high a γ as we can for a reasonable computer cost. For all of the traits this corresponds to a γ between 0.0005 and 0.005. The tendency is that a higher dimensionality means a higher rejection rate but this is not always the case with the four allele trait Serum Es being the most computer costly.

We have also investigated the number of Gibbs sweeps per iteration needed and the number of EM iterations needed to get good estimates. Obviously the higher we can make either of these parameters, the greater will be the accuracy of the estimates. We have found that for fixed computer cost, greater accuracy is achieved by putting greater emphasis on the number of Gibbs sweeps per iteration, rather than the number of iterations.

Increasing the dimensionality seems to imply that a greater number of iterations are need to achieve convergence of the Gibbs EM algorithm. Most of the traits seem to settle down and show no change after about 250–500 iterations, however, the six allele trait seems to need more than 1000 iterations to satisfy us that it has settled on the point of convergence. For proper MCMC convergence we need to increase the number of Gibbs sweeps in some way as the algorithm progresses, but have found that practically for all traits we have found that after a 5% burn-in period, 1000 EM iterations of 10,000 Gibbs sweeps per iteration gives accuracy almost to four decimal places. An increase in dimension shows up the differences between 1000 and 10,000 Gibbs sweeps per iteration. We have more confidence in the answers produced by the 10,000 sweep regime but the differences are small enough to be of little importance in practice when computer cost may be a more important factor.

Biological Implications

We have shown that the Gibbs Sampler can provide us with accurate estimates of probabilities for multi-allele traits on the PH pedigree. Consequently, we used the method to try and answer the simple biological question about the different management programs which initiated this thesis for the three remaining traits which cannot be analysed with the peeling process.

The results are similar to those found in §2.3 for the two and three allele traits. There is

a huge variety of allele frequencies for these multi-allele traits in the founding horses which is indicative of a large amount of interbreeding between domestic and Przewalski horses in the population from which the founders were taken. Because of this large variation, results for the three groups containing one of the interesting horses are no more unusual than the other groups. This implies that the management programs which are based on so-called 'pure' and 'contaminated' Przewalski horses and which involve an additional mild form of selection, are not justified as all of the founders contained varying amounts of domestic horse genes and selection against these genes, whilst being beneficial in the short term, may be doing unknown damage to the genome on a long term basis. Further evidence of this is shown when we compare the founding and current population frequency estimates. For the three multi-allele traits examined in this chapter, all show a dangerous drop in the frequency of the rarest allele, to such an extent that the allele is in real danger of disappearing from the population altogether. The biological effects of such disappearances are unknown.

Chapter 4

Full Bayes Approach

Clearly, bins are for the birds!
(J Tukey)

4.1 Introduction

In the previous chapters we introduced an EM algorithm as an efficient way of finding the MLEs of the allele frequencies of the founders which are needed for accurate ancestral inference. We initially used the peeling process in the EM algorithm but due to computational restrictions had to develop a simulation method using the MCMC algorithm, the Gibbs Sampler for three and higher allele traits. We found that for the Przewalski horse pedigree, even for the six allele system, the MCMC approach converged quickly. The results from the simulation were checked with the exact results for the two and three allele traits and were found to be very good. The process was then continued for higher allele traits for which no exact answers were available. Convergence was achieved and with the use of multiple starting points confidence was increased that the convergence was to the MLEs.

However, there are still problems with the methods. The algorithm can produce good MLEs but the parameters that are needed in the run - the relaxation parameter γ, the number of iterations, and the number of sweeps per iteration in the Gibbs EM algorithm all need to be investigated and adjusted in each case. Importantly, time has to be spent checking for multi-modality. This is relatively cheap computationally but still needs to be repeated many times with different starting frequencies to create greater confidence in any convergence. Despite all of our checking, there is still doubt as to whether we have sampled from all relevant sub-spaces and consequently converged to the correct place.

Other authors, such as Geyer and Thompson (1995), have found that convergence can take ridiculous amounts of time and this has been the reason for alternative sampling methods using the Gibbs Sampler such as auxiliary variables, Metropolis Coupled Markov Chains and simulated tempering, which are highlighted in §5.1.2.

With its origins in Bayesian statistics, the Gibbs EM algorithm can be thought of as pseudo-Bayes methods. Can we continue using the Bayesian framework to find a different method of estimating the allele frequencies, π_is? The Bayesian approach requires a prior on π, so instead of using an EM algorithm, we could place a prior on π and then use the Gibbs Sampling to sample from the posterior distribution of π. In fact, with a careful choice of prior this posterior distribution can be used for non-Bayesian maximum likelihood estimation. This is not fully Bayesian inference, but we are using the Bayesian framework as a useful computational tool.

4.2 Bayesian Gibbs Algorithm

In §2.2 we showed that as we are not changing the penetrance or transmission probabilities in our genetic model then we can write the probability as

$$L\{\phi(D), \pi_F\} = P\{\phi(D) \mid \pi_F, G\} \tag{4.1}$$

where D is the set of individuals in the pedigree whose state with respect to the trait of interest has been observed, $\phi(D)$ is the set of phenotypes for those individuals, π_F is the founder allele frequencies and G is the pedigree structure. However, we can go further because if we let $\pi_F = (X_1, \ldots, X_k)$ where X_k denotes the founding population frequency of allele k

$$\begin{aligned} L\{\phi(D), X_1, \ldots, X_k\} &= P\{\phi(D) | X_1, \ldots, X_k, G\} && (4.2) \\ &= \sum_{i_1} \cdots \sum_{i_n} x_1^{\alpha_1} \ldots x_k^{\alpha_k} \prod \tau(i_k, i_{f_k}, i_{m_k}) \prod \rho\{\phi(l), i_l\} && (4.3) \end{aligned}$$

where α_i for $i = 1$ to k is the number of occurrences of allele i in the founding population for given $i_1, \ldots, i_n$.

If we now put a prior, p, on $X_1, \ldots, X_k$, this implies a joint distribution for $X_1, \ldots, X_k$ and all phenotypes of

$$P\{X_1 = x_1, \ldots, X_k = x_k, \phi(D)\} = p(x_1, \ldots, x_k) \sum_{i_1} \cdots \sum_{i_n} x_1^{\alpha_1} \ldots x_k^{\alpha_k} \prod \tau(i_k, i_{f_k}, i_{m_k}) \prod \rho\{\phi(l), i_l\}. \tag{4.4}$$

We can now use MCMC to sample from the conditional distribution of $X_1, \ldots, X_k$ given that observed phenotypes are equal to the observed values. For this,

$$P\{X_1 = x_1, \ldots, X_k = x_k \mid \phi(D)\} \propto p(x_1, \ldots, x_k) \sum_{i_1} \cdots \sum_{i_n} x_1^{\alpha_1} \ldots x_k^{\alpha_k} \prod \tau(i_k, i_{f_k}, i_{m_k}) \prod \rho\{\phi(l), i_l\}. \tag{4.5}$$

Now, we can write

$$P\{X_1 = x_1, \ldots, X_k = x_k \mid \phi(D)\} \propto p(x_1, \ldots, x_k) \times L\{\phi(D), x_1, \ldots, x_k\}. \quad (4.6)$$

We can get full information on the posterior distribution by MCMC sampling and, by using Equation (4.6), we can maximise the $L\{\phi(D), x_1, \ldots, x_k\}$ term to get MLEs for $\pi_F = (X_1, \ldots, X_k)$.

The remaining problem is what to choose for the prior distribution on $p(x_1, \ldots, x_k)$.

4.2.1 Dirichlet Distribution

We have chosen a *Dirichlet* distribution. It is convenient and is the conjugate prior distribution for the parameters of the *Multinomial* distribution. The *Dirichlet* distribution is a multivariate generalisation of the *Beta* distribution.

A continuous random vector $\mathbf{x} = (x_1, \ldots, x_k)$ has a *Dirichlet* distribution of dimension k, with parameters $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_{k+1})$, where $(\beta_i > 0,\ i = 1, \ldots, k+1)$ if its probability density $Di_k(\mathbf{x} \mid \boldsymbol{\beta})$, for $0 < x_i < 1$ and $x_1 + \ldots + x_k < 1$, is

$$Di_k(\mathbf{x} \mid \boldsymbol{\beta}) = c x_1^{\beta_1 - 1} \ldots x_k^{\beta_k - 1} (1 - \sum_{i=1}^{k} x_i)^{\beta_{k+1} - 1}, \quad (4.7)$$

where

$$c = \frac{\Gamma(\sum_{i=1}^{k+1} \beta_i)}{\prod_{i=1}^{k+1} \Gamma(\beta_i)}. \quad (4.8)$$

If $k = 1$, $Di_k(\mathbf{x} \mid \boldsymbol{\beta})$ reduces to the *Beta* density $Beta(x \mid \beta_1, \beta_2)$

The marginal distribution of $\mathbf{x}^{(m)} = (x_1, \ldots, x_m)$, $m < k$, is the *Dirichlet*

$$p(\mathbf{x}^{(m)}) = Di_m(\mathbf{x}^{(m)} \mid \beta_1, \ldots, \beta_m, \sum_{j=m+1}^{k+1} \beta_j). \quad (4.9)$$

The conditional distribution, given $x_{m+1}, \ldots, x_k$, of

$$x_i' = \frac{x_i}{1 - \sum_{j=m+1}^{k} x_j}, \quad (4.10)$$

for $i = 1, \ldots, m$ is also *Dirichlet*, $Di_m(x_1', \ldots, x_m' \mid \beta_1, \ldots, \beta_m, \beta_{k+1})$. In particular

$$p(x_i' \mid x_{m+1}, \ldots, x_k) = Beta(x_i' \mid \beta_i, \sum_{j=1}^{m} \beta_j + \beta_{k+1} - \beta_i),\ i = 1, \ldots, m. \quad (4.11)$$

For this problem, we have chosen to use a prior of *Dirichlet*, $\beta_i = 1$ for $i = 1, \ldots, k+1$. Hence, $p(x_1, \ldots, x_k)$ is constant, so MLE is x for which the posterior distribution is maximised.

4.2.2 Dirichlet Sampling

The standard approaches to sampling from a *Dirichlet* distribution rely on generalising drawing samples from a *Beta* distribution. There are many approaches to sampling from a *Beta* distribution with one of the most popular given in Ripley (1987).

Algorithm 4.1. BETA:

1. *Generate* $U_1, U_2 \sim U(0,1)$.
2. *Let* $V_1 = U_1^{1/\alpha}, V_2 = U_2^{1/\beta}$.
3. *Let* $W = V_1 + V_2$.
4. *If* $W \leq 1$ *then return* V_1/W, *else repeat from 1.*

□

However, the rejection sampling which underpins this method becomes computationally prohibitive for large a and b. As we are interested in traits up to 6 alleles, we have to sample from a *Beta*(6,6) which gives rise to a large rejection rate.

A better method for generating from a *Beta* with integer values a and b is given in Ahrens and Dieter (1974).

Algorithm 4.2. JÖHNK'S BETA:

1. *Generate* $U_1, U_2, \ldots, U_n \sim U(0,1)$ *where* $n = a + b - 1$.
2. *Deliver the* a^{th} *smallest of the* U_i *as the sample* x *from the Beta*(a,b) *distribution.*

□

So using Jöhnk's Beta algorithm for simulating from a *Beta* we use the following stick breaking algorithm to simulate from a *Dirichlet*.

Algorithm 4.3. DIRICHLET SAMPLING:

1. *Simulate* x_1 *from a Beta*$(\beta_1, \sum_{i=2}^{k+1} \beta_i)$ *distribution.*
2. *For* $j = 2, \ldots, k-1$, *simulate* ϕ_j *from a Beta*$(\beta_j, \sum_{i=j+1}^{k+1} \beta_i)$ *distribution, and let* $x_j = (1 - \sum_{i=1}^{j-1} x_i)\phi_j$.
3. *Finally, set* $x_k = 1 - \sum_{i=1}^{k-1} x_i$.

□

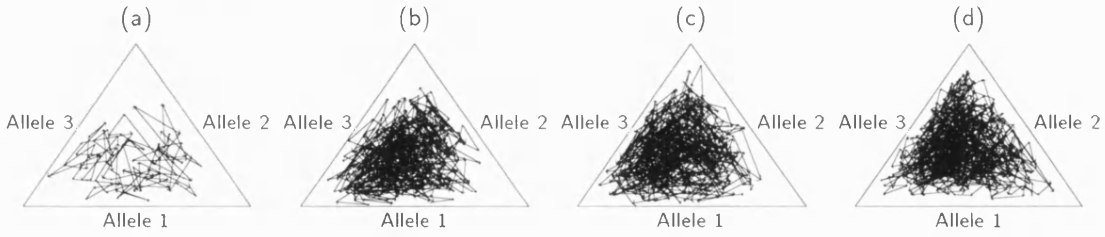


Figure 4-1: PLOTS OF BAYES GIBBS SAMPLER FOR TRAIT C71: Figure (a) shows the first 100 draws from the Bayes Gibbs sampler algorithm. The arrows indicate the direction of the draw and the size of the arrow head indicates the length of the draw. Figures (b)–(d) show 1000 draws in the algorithm after 1000, 50,000 and 99,000 draws have been taken.

4.2.3 Methodology

We are going to use the Gibbs Sampler to generate samples from the posterior distribution, which as we have chosen a constant prior distribution, is equivalent to sampling from the likelihood distribution. By repeated sampling, we can build up an estimate of the likelihood surface and compare them with the exact surfaces shown in Figure 2-2.

In implementing the algorithm we must firstly check that the sampling procedure is visiting all regions of the sample space. Figure 4-1 shows the results for the allele frequencies for trait C71 for a run of 100,000 consistent draws of the Bayes Gibbs algorithm. The results are represented in a triplot format where the allele frequencies are represented as the perpendicular distances from the edges of a equilateral triangle. Figure (a) shows the first 100 draws of the algorithm; Figures (b) - (d) show 1000 draws of the algorithm after the process has been running for 1000, 50,000 and 99,000 draws respectively.

Ideally the algorithm should have settled down after a few thousand draws and it should explore *all* regions of the sample space. From the plots there appears to be problems of sampling from the corners, which are where any one allele's frequency is large. Similarly, few values are taken along the edges of the space, which is where one of the frequencies is small. However, the MLEs for the C71 trait are (0.2687, 0.4510, 0.2803) and so are not highly weighted in favour of one of the alleles. Consequently, we would not expect many draws giving a very small or a very large allele frequency for one of the alleles. So, the sampler appears to be sampling from the correct places. There appears to be little difference between the plots and so the algorithm appears to have settled down quickly.

From these sets of allele frequencies it is easy to calculate in a Bayesian analysis, an estimate of the marginal posterior means and their standard deviations. However, we are also interested in the MLEs and the posterior surfaces which raises the question of how to find estimates of the surface from the MCMC realisations. We have discussed and given a brief review of density estimation in §1.15 and the coming sections use a simple histogram and Gaussian kernel. However, in Chapter 5 we outline possible avenues of further work involving a Rao-Blackwell technique cited in Gelfand and Smith (1990).

4.3 The Two Allele Traits

In this section we use this Bayes Gibbs sampling to get estimates of the likelihood of the two alleles and compare them with the exact likelihood curves.

Updating the prior is computationally very cheap and it is the Gibbs sweeps to obtain a consistent configuration that take the majority of the time. Consequently, it is the γ parameter which again mostly affects the run-time.

For the Gibbs part of the algorithm we use the parameters we found in Chapter 3 that gave the best balance between accuracy and computer cost. For each of the two allele traits, we obtain 100,000 consistent configurations using a γ of 0.005.

Results

Point Estimates

We have constructed a new algorithm designed to provide us with samples from the posterior and by making the prior distribution constant have made it possible to do non-Bayesian analysis. In this section we compare results from this new algorithm with exact results obtained from the likelihood approach.

Using a Bayesian framework, the new algorithm sets up a posterior distribution

$$P\{X_1 = x_1, \ldots, X_k = x_k, \mid \phi(D)\} = c\ p(x)p(\phi(D) \mid X = x) \tag{4.12}$$

and this is sampled from using a Gibbs Sampler. From this method it is easy to find $E(X_i)$ and $SD(X_i)$ and the results of the rarer allele for each two allele trait from 100,000 realisations of the algorithm are shown in Table 4.1 along with the rejection rate and run time of the algorithm.

To get similar estimates from the likelihood approach, we use the peeling algorithm which directly calculates $p(\phi(D) \mid X = x)$ and then for a two allele trait, where $X_1 + X_2 = 1$, $E(X_1)$ when $X \sim c\ p(x)p(\phi(D) \mid X = x)$ is

$$E(X_1) = c \int_0^1 p(x)p(\phi(D) \mid X = x)\ x_1\ dx_1 \tag{4.13}$$

$$= \frac{\int_0^1 p(x)p(\phi(D) \mid X = x)\ x_1\ dx_1}{\int_0^1 p(x)p(\phi(D) \mid X = x)\ dx_1}. \tag{4.14}$$

So for each of the two allele traits, we take 101 allele frequencies ranging between 0 and 1 for allele 1, hence fixing the frequency of allele 2. At each frequency we use the peeling algorithm to calculate the likelihood. We then transform the data into a likelihood distribution, ensuring that we scale by the correct amount, and then use Simpson's rule to get an estimate of the mean; the method enables us get estimates of the standard deviation as well by calculating $E(X_i^2)$. The results for the rarer allele for each of the two allele traits is also shown in Table 4.1. Although we have not been interested in the $E(X_i)$ in previous chapters, it is the easiest estimate to use for comparison of the simulated results

with the exact answers.

Point Estimates of the Rarest Frequency: Comparison of Exact and Simulated Means Using 100,000 Bayes Gibbs Draws							
Trait	Exact Results		Simulated Results				Effective
	Mean	SD	Mean	SD	Rej Rate (%)	CPU Time (Sec)	*Bin* Sample Size
ALB	0.1903	0.119	0.1888	0.118	41	563	11.4
BF	0.1837	0.114	0.1827	0.114	48	630	11.8
CA1	0.1843	0.115	0.1839	0.114	67	933	11.9
GPI	0.2083	0.128	0.2065	0.125	37	441	10.8
Hb	0.3451	0.164	0.3417	0.162	33	542	8.6
MPI	0.2643	0.141	0.2635	0.140	31	402	10.0
PEPB	0.3911	0.171	0.3920	0.168	33	496	8.4
PEPD	0.2822	0.131	0.2814	0.134	96	6611	11.3
PGM1	0.2547	0.141	0.2535	0.136	88	2175	10.4

Table 4.1: Point Estimates for the Two Allele Traits: The table displays the exact marginal mean of the rarest allele frequency of the two allele traits and estimates of the marginal mean derived from 100,000 iterations of the Gibbs Sampler. Estimates of the standard deviation are also shown along with the rejection rate and computer run time. The last column presents the effective *Binomial* sample sizes given the posterior mean and standard deviation.

If we compare the exact and simulated marginal means, we can see that using 100,000 realisations from the Bayes Gibbs algorithm gives us a good estimate of the true means. All of the results are within 0.002, with the largest discrepancy of 0.0018 shown in the GPI trait. The estimates of the SDs are also very good but we also notice that the general size of the error estimates is quite large, in keeping with the small number of founders (13). The last column of Table 4.1 presents the effective binomial sample sizes given the marginal posterior means and standard deviations for the nine traits, which range from approximately 8 - 12. For example, the effective sample size of 10.0 for the trait MPI means that the indirect information on the founder allele prevalence for this trait from the 244 horses in Figure 2-1 is equivalent to a directly observed *IID Bernoulli* sample on this trait of size 10. Further details on effective sample size will be given in §4.5.

Looking at the run parameters we can see that as we have kept γ and the total number of consistent realisations identical for each trait, the rejection rate depends on the trait itself and there is quite a difference. For example, for trait MPI the 100,000 consistent realisations take approximately 150,000 draws from the process giving a rejection rate of approximately 30% and are made in approximately seven minutes. For trait PEPD, because of the set up of the pedigree for this trait, the 100,000 consistent realisations take approximately two million draws giving a rejection rate of 96%. The run time is consequently longer at approximately two hours.

Effect of Frequency with which $X_1, \ldots, X_k$ are updated in the Gibbs Sampler

The estimates of the $E(X_i)$s were obtained by repeatedly using the Gibbs Sampler to make draws from the likelihood distribution. This was done by making a draw from the prior information on $X_1, \ldots, X_k$ and then for each draw sweeping through the pedigree once with a Gibbs Sampler. Estimates of the $E(X_i)$s can then be easily extracted from these estimates. Does increasing the number of Gibbs sweeps per prior update increase the accuracy of the estimates? To investigate this we use trait Hb.

Table 4.2 shows the estimates of the marginal mean of the rarer allele and the associated SD for 100,000 consistent realisations from the likelihood distribution but with increasing numbers of Gibbs sweeps per prior update. The run time is also included.

Point Estimates for Different Numbers of Gibbs Sweeps Per Prior Update			
Gibbs Sweeps Per Prior Update	Mean	SD	CPU Time (Secs)
1	0.3417	0.162	542
2	0.3437	0.162	987
5	0.3448	0.161	2967
10	0.3440	0.161	4751
100	0.3429	0.161	46250
Exact	0.3451	0.164	-

Table 4.2: Effect of Number of Gibbs Sweeps per Prior Update: The table displays estimates of the marginal mean of the rarest allele frequency for trait Hb derived from 100,000 iterations of the Gibbs Sampler. In each case a different number of Gibbs sweeps per prior update is used. Estimates of the standard deviation are also shown along with the computer run time.

From the table we can see that increasing the number of Gibbs sweeps per prior update has little effect on the marginal mean estimates and their associated standard deviation. The only consequence is the approximately linear increase in run time. This implies that for a fixed computer cost it would be better to use only one Gibbs sweep per prior update and so achieve more realisations from the likelihood improving accuracy. For the remainder of this chapter, we will only use one Gibbs sweep per prior update.

Estimate of the Likelihood Surface

So far in this thesis we have been interested in the MLEs and not $E(X_i)$, which have been used in this section so far. Obtaining MLEs means getting an estimate of the likelihood surface from the realisations produced by the Bayes Gibbs algorithm which will require some form of density estimation. Figure 4-2 compares two estimates of the likelihood surface with the exact likelihood for the rarer allele for the two allele traits Hb and MPI. 100,000 realisations of the Bayes Gibbs algorithm are taken for each of the traits and used in the estimates. The first estimate uses a histogram with 40 bins and some splines to make a continuous estimate. The second uses a Gaussian kernel estimate with a grid of

40 points and window width h, 0.07. The window is centred at each data point and the contributions from each of the realisations are summed. The algorithm is very efficient, using the FORTRAN language to rattle through the summation. The process also uses a cutting procedure which assumes that the contribution of any point that is $> 4h$ from one of the grid points is negligible. The exact likelihood surface is shown for comparison.

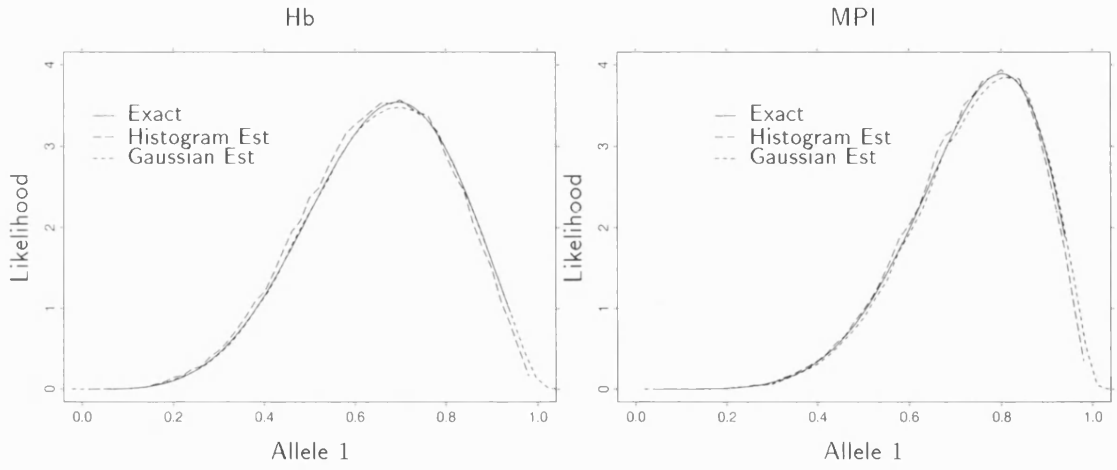


Figure 4-2: Comparison of Likelihood Surfaces for Two Allele Traits: This figure compares the exact likelihood surface with two estimates using the Bayes Gibbs algorithm for the rarer allele for the two allele traits Hb and MPI. The exact curve was obtained by transforming the curves in Figure 2-2. The estimates were derived from 100,000 realisations of the Gibbs Sampler and the curves are shown for a histogram estimate using 40 bins and an estimate using a Gaussian kernel at 40 data points with bandwidth 0.07.

The simulated surfaces are very promising. Even the simple density estimate using a histogram gives a very good fit when compared to the exact likelihood curve. The estimate is slightly improved and smoother when a Gaussian kernel is used, although the inherent underestimation of the mode when using a density estimate is shown.

Both sets of results – the $E(X_i)$s and the surface estimates – confirm that the Bayes Gibbs sampling procedure appears to work efficiently for the two allele case. Emphasis has changed however. With the Gibbs EM algorithm, the cost was in computing many consistent Gibbs simulations but convergence to the MLEs was obtained directly from the process; here the cost of generating the Gibbs samples using the Bayesian framework is relatively cheap and so the $E(X_i)$s can be quickly calculated. However, we are also interested in obtaining the MLEs and this needs some form of density estimation which is going to be costly, especially as the number of dimensions increases when more points are going to be required.

4.4 The Three Allele Trait - C71

We now extend the Bayes Gibbs algorithm to the three allele traits and concentrate on trait C71.

4.4.1 Correlation Plots

A worry about the Bayes Gibbs sampling procedure that we have defined is how correlated successive draws are. As in Chapter 3, we use a relaxation parameter to guarantee that the Markov Chains for sampling traits with three or more alleles are irreducible and it is this relaxation parameter which affects the correlation. To see its effect we can use a plot of the *autocorrelation*.

Definition 4.1. The *autocovariance* function at lags for a time series $X_t, t = 1, \ldots, N$, can be defined to be

$$c_s = \sum_{t=1}^{N-s} \frac{(x_t - \bar{x})(x_{t+s} - \bar{x})}{N}. \tag{4.15}$$

Definition 4.2. The *autocorrelation* function is defined to be

$$r_s = \frac{c_s}{c_0}. \tag{4.16}$$

A plot of the autocorrelation against the lag is called the *correlogram* and it is this which will tell us how correlated successive consistent realisations from the Markov Chain are for different lags. We concentrate only on allele 1 for trait C71 as the other alleles show similar results. We firstly run the process to get 100,000 consistent realisations with a $\gamma = 0.005$. The correlogram for lags up to 125 is shown in Figure 4-3(b).

From the plot we can see that the draws are correlated up to a lag of about 50. After this, the process settles down but does show a surprisingly large number of significant higher lags. This indicates that a consistent pedigree configuration is correlated with approximately the previous 50 consistent configurations. This implies that we should be very careful and do all we can to verify any results obtained from the sampling process. We can certainly reduce the correlation by increasing γ to as high as possible given computational time restrictions.

We run the sampling algorithm for 100,000 consistent draws for four γ parameters: 0.01, 0.005, 0.0005 and 0.00005. The run parameters are given in Table 4.3.

Run Parameters for the Bayes Gibbs Algorithm for Different γ Parameters		
γ ($\times 10^{-2}$)	Rejection Rate (%)	CPU Run Time (Secs)
1	88	3600
0.5	66	1320
0.05	10	580
0.005	1	500

Table 4.3: Run Parameters of the Bayes Gibbs Algorithm: The table shows the run parameters for the Bayes Gibbs algorithm for trait C71 for four different γ parameters keeping the number of consistent realisations fixed at 100,000.

For low relaxation parameters, $\gamma = 0.00005$, almost every realisation from the sampling

process will be taken, whereas for high relaxation parameter, $\gamma = 0.01$, approximately only one consistent realisation for every 10 realisations will be taken. In addition to these parameters we have plotted the correlogram up to lags of 125 for the four runs. These are shown in Figure 4-3.

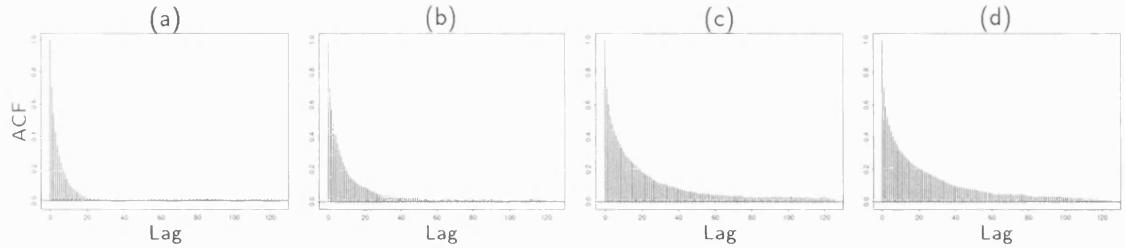


Figure 4-3: CORRELOGRAMS FOR TRAIT C71: Plots (a) - (d) show the correlograms up to lag 125 for 100,000 draws of allele 1 for trait C71 for four relaxation parameters: 0.01, 0.005, 0.0005 and 0.00005.

Results

The results from Table 4.3 confirm that the process is computationally cheaper than the Gibbs EM algorithm. In §3.4.1, where we investigated the effect of varying γ for 10 EM iterations using 10,000 Gibbs sweeps per iteration, a γ of 0.01 gave a rejection rate of approximately 95% and a run time of approximately three hours. Here we can see that a similar 100,000 consistent realisations gives a rejection rate of approximately 88% and takes only an hour.

The plots of the correlograms for the four γ parameters show that the correlation dies away at a lower lag for higher γ. For example, Figure 4-3(a) is the correlogram for $\gamma = 0.01$ and the lag appears to die away at approximately 20, whereas for a γ of 0.00005, shown in Figure 4-3(d), the lag doesn't die away until 120. This shows, as expected, a higher γ reduces the correlation.

As a result of these findings and despite the additional computational cost, it would appear to be worthwhile increasing the relaxation parameter to 0.01 when running the Bayes Gibbs sampling algorithm on trait C71. However, what is the effect on parameter estimates and for a fixed run time can we optimise the choice of γ?

Optimising γ and Run Time

Partial ACF

We would expect the serial correlation of the random samples for a given parameter from the Bayes Gibbs process to be modelled approximately by an AR(1) process. However, it is tricky to estimate the order of the AR process using the ac.f alone. A better aid to determine the order of the AR process is the *partial* ac.f. The partial ac.f of order 2 measures the excess correlation between X_t and X_{t+2} not accounted for by the lag 1 effect;

similarly, the partial ac.f of order 3 measures the excess correlation between X_t and X_{t+3} not accounted for by the lag 1 and lag 2 effect. The partial ac.f for the 100,000 realisations of the Bayes Gibbs algorithm on trait C71 using a γ of 0.01 is shown in Figure 4-4.

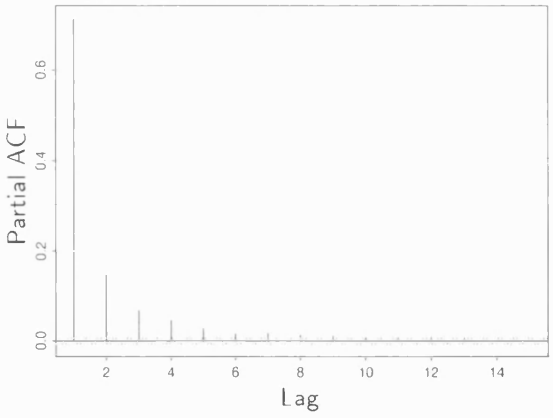


Figure 4-4: PARTIAL AC.F: The figure shows the partial ac.f plot up to lag 15 for 100,000 realisations of the Bayes Gibbs algorithm for allele 1 for trait C71 using a γ of 0.01.

There is a non negligible partial ac.f(2) but the values die away rapidly and so let us assume that the Bayes Gibbs process is approximately modelled by an AR(1) process. Now for an AR(1) time series

$$SE_{AR(1)}(\bar{Y}) = SE_{IID}(\bar{Y}) \sqrt{\frac{1+\rho_1}{1-\rho_1}} \tag{4.17}$$

$$\approx \frac{s}{\sqrt{n}} \sqrt{\frac{1+\hat{\rho}_1}{1-\hat{\rho}_1}} \tag{4.18}$$

where ρ_1 is the first-order autocorrelation. So we could be able to optimise the choice of γ on run time because as γ decreases so does run time but ρ might increase.

To investigate this let us re-run the Bayes Gibbs algorithm for two numbers of realisations – 10,000 and 100,000 – for different values of γ. In addition to the run parameters given in Table 4.3 we note the estimates of ρ_1 and the standard deviation. The results are shown in Table 4.4.

Results

We initially ran the algorithm for 10,000 realisations but found that the estimates of $\hat{\rho}_1$ do not change much for decreasing γ. We increased the number of realisations to 100,000 but the same effect was observed. So the optimisation that we postulated does not materialise.

However, it is interesting to note the effect on the standard deviation of decreasing γ. For 100,000 realisations, the estimates obtained are very close across the range 0.0005 to 0.01 and of course, for a fixed computer cost, we can achieve many more realisations for low γ which increases the accuracy of any estimates. It appears that running the Bayes Gibbs algorithm with as high a γ as possible is not so important. However, Figures 4-3

Optimising γ for Trait C71						
10,000 Realisations from Bayes Gibbs Algorithm						
γ	Rej Rate	CPU Time	Allele 1		Allele 2	
($\times 10^{-2}$)	(%)	(Sec)	SD	$\hat{\rho}_1$	SD	$\hat{\rho}_1$
1.0	88.01	359	0.1545	0.7177	0.1608	0.7142
0.9	85.59	299	0.1532	0.7082	0.1607	0.7138
0.7	76.52	190	0.1540	0.7000	0.1580	0.7017
0.5	65.93	134	0.1596	0.7200	0.1649	0.7338
0.3	48.10	94	0.1549	0.7023	0.1568	0.7069
0.1	18.83	62	0.1526	0.6983	0.1603	0.7157
100,000 Realisations from Bayes Gibbs Algorithm						
γ	Rej Rate	CPU Time	Allele 1		Allele 2	
($\times 10^{-2}$)	(%)	(Sec)	SD	$\hat{\rho}_1$	SD	$\hat{\rho}_1$
1.0	87.97	3617	0.1542	0.7097	0.1615	0.7167
0.5	65.45	1656	0.1548	0.7099	0.1614	0.7203
0.1	18.94	618	0.1554	0.7115	0.1606	0.7151
0.05	10.21	569	0.1556	0.7143	0.1606	0.7171

Table 4.4: Optimising γ and Run Time for Trait C71: The table displays parameters of the Bayes Gibbs process for trait C71 for decreasing γ. Two number of realisations have been used – the top half of the table uses 10,000 realisations and the bottom half of the table uses 100,000 realisations. For each run, the rejection rate, CPU time, estimates of the standard deviation and $\hat{\rho}_1$ are shown for allele 1 and allele 2.

show that the longer lags do decrease and this is more important than the $\hat{\rho}_1$. We conclude that ideally we should run the algorithm with as high a γ parameter as possible.

For the next few sections we use the same set of data. 100,000 consistent samples are taken for the trait C71 using a γ of 0.01 which takes approximately one hour and investigate methods of obtaining estimates of the likelihood surface using different density estimation techniques.

4.4.2 Histograms

The first density estimate is the 2D histogram. We take the 100,000 simulations of the Bayes Gibbs algorithm and bin them. Figure 4-5 shows the histograms obtained for four different bin widths: 10, 20, 50, and 100 bins in each of the axis directions.

The results are what we would expect. For a low number of bins we achieve the shape of the exact density, shown in Figures 2-3(a) and (b), but trying to estimate the MLEs of this distribution will not be accurate. For increasing number of bins we effectively smooth the estimate until at 100 in each direction we have oversmoothed for the number of simulations and start to lose the general shape of the underlying distribution. So using 20×20 bins appears to give the best results.

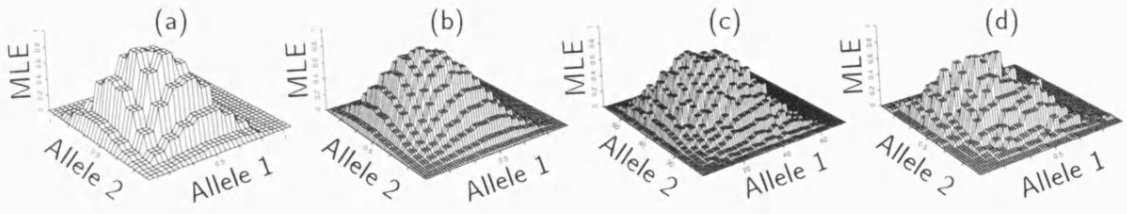


Figure 4-5: 2D Histograms for Trait C71: This figure shows two dimensional histograms from 100,000 simulations for trait C71. Figures (a)–(d) show the histograms where the simulations of the allele frequencies are binned into 10, 20, 50 and 100 bins on each axis. For bins 50 and 100 only every 2^{nd} and 5^{th} line was printed.

4.4.3 Interpolation

We shall look at a way of obtaining a smooth density estimate of the likelihood surface from these histograms using interpolation.

Method

The method we shall use is from Akima (1978). The method was developed for z values given at points irregularly distributed in the $x - y$ plane, but we can use it to provide a different number of grid points at which to calculate z values than the number of bins for which we have z values.

The method divides the $x - y$ plane into triangular cells, each having projections of three data points in the plane as its vertices. The interpolating function between each triangle is a fifth degree polynomial in x and y.

The method can be extended by using partial derivative information which hopefully gives better smoothing. The algorithm works by taking a certain number of points, n_c, that are closest to the data point at which derivative information is required and using the sum of vector products of the vectors joining these points, getting new values for the co-ordinates of the data point and its corresponding z value. The choice of n_c is left to the user. Obviously $n_c \geq 2$ and must be less than the total number of data points, although in practice, the upper limit is dictated by computational cost.

Different Interpolation Routines

We use the above interpolation procedure on the 100,000 consistent Bayes Gibbs realisations of trait C71. Initially, we bin the data using a histogram of 10 by 10 bins. We then use four different methods to produce a smooth estimate of the surface. Firstly, we use the interpolation routine using the same 10 by 10 grid. Secondly, we use the interpolation routine to increase the grid to 40 by 40 but use no partial derivative information in making the smooth surface estimate. Thirdly, we use $n_c = 2$ to take account of local partial derivative information to generate our estimate of the surface and fourthly, use n_c

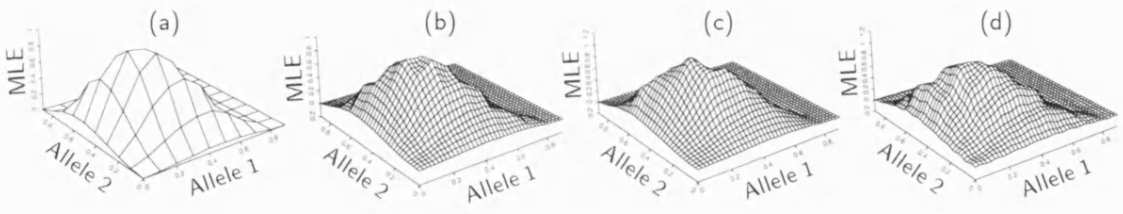


Figure 4-6: INTERPOLATION PLOTS FOR 10 × 10 HISTOGRAM: The figure contains four plots. (a) is a perspective plot of the raw data which has been binned into 10 by 10 bins. (b) is a perspective plot which uses an interpolation method to generate a grid of 40 by 40 points from these initial bins. (c) is a perspective plot which uses this 40 by 40 grid but also uses gradient information at two neighbouring points to smooth the surface. (d) is the same as the (c) but uses a greater number of neighbouring points for gradient information.

= 25, which is the largest number of neighbouring points that the routine can handle for a reasonable computer cost. The surfaces obtained for each of these four procedures are shown in Figure 4-6.

Results

Figure 4-6(a) is the smoothed surface estimate obtained from the histogram in Figure 4-5(a) which undersmoothes the surface. The general shape of this estimated likelihood surface compares favourably with the exact likelihood surface for trait C71 shown in Figures 2-3(a) and (b) but the small number of grid points will make any ML calculation inaccurate. The effect of this undersmoothing is exaggerated in (b) where we use the interpolation technique to increase the number of grid points. Plots (c) and (d), where gradient information is used, appear to give no improvement to the estimated surface.

Consequently, we conclude that using the interpolation routine to generate a grid of 40 × 40 bins using no gradient information gives the best estimate of the surface, while noting that this example suffers from the in-built undersmoothing from the initial 10 × 10 histogram.

Different Bins Sizes

Using this technique to obtain a smooth density estimate of the surface, we can now obtain smooth estimates from the other histograms in Figure 4-5 with 20, 50 and 100 bins in each direction. So for each of the different histograms we use the interpolation routine with no partial derivative information to provide a grid of 40 × 40 points from which we create the smooth estimate of the surface. This involves increasing the number of grid points for the 20 × 20 bin histogram and reducing the number of grid points for the 50 and 100 bin histograms. For comparison, the result for the 10 × 10 histogram is shown alongside the results for the three others in Figure 4-7. The top line shows the perspective plots of the surface estimate and the lower layer shows the respective contour plots.

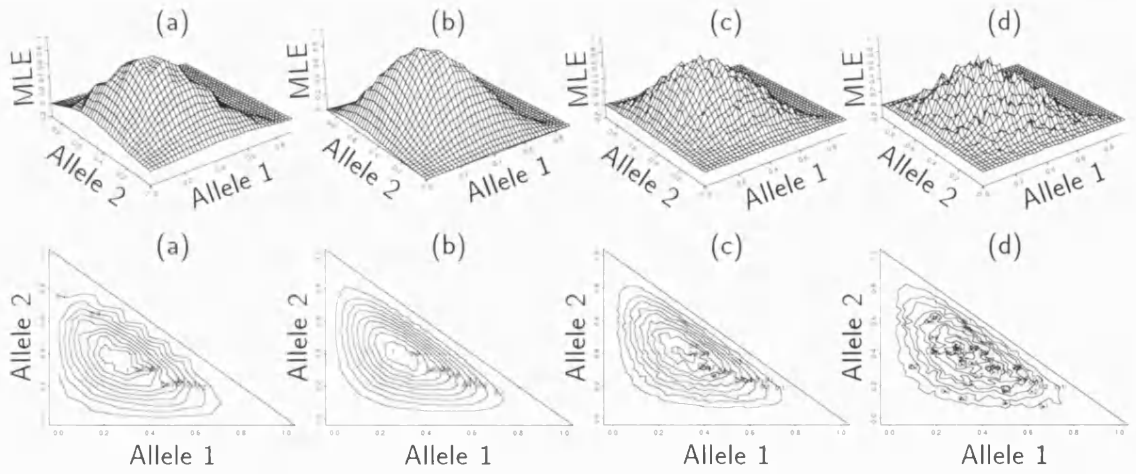


Figure 4-7: Interpolation Plots for Different Bin Widths: This figure shows two dimensional plots for 100,000 realisations of the Bayes Gibbs algorithm for trait C71. Figures (a)-(d) on the top line show perspective plots of the estimate of the likelihood surface. The realisations are initially binned into 10, 20, 50 and 100 bins on each axis respectively. These are used in the interpolation function where a grid of 40 $\times$ 40 and no partial derivative information is used. The bottom line shows the respective contour plots of the estimated surface.

Results

The process again shows the effect of oversmoothing. For a grid of 10 $\times$ 10 bins we achieve the general shape of the likelihood surface but it is undersmoothed. For 20 $\times$ 20 bins the surface appears nicely smooth, but for high numbers the effect of the oversmoothing in the histograms is apparent and the interpolation routine cannot undo the spiked estimate.

We can also see that the apparent position of the MLEs, which is represented by the 1 on the contour plots, is highly susceptible to the number of bins. This probably means that we should increase the number of simulations to get a better estimate of the MLEs.

4.4.4 Gaussian Kernel

In §4.3 we found that a better estimate of the likelihood curves of some two allele traits was obtained using a Gaussian kernel. In 2D we can think of dropping a bivariate normal on the surface to be estimated at a grid of points. Because the normal distribution has bounds from $-\infty$ to ∞, for each grid point we should add up all the contributions from this 2D Gaussian kernel for all the remaining points. The easiest way to do this is by storing the contributions in a matrix. This gives rise to huge amounts of data storage and even if we restrict the algorithm to look at points less than $4h$ from the grid point, as anything larger than this has a negligible contribution, it is still computationally very costly.

A popular package to perform 2D Gaussian kernel density estimation is Maechler

and Nason's `kde2D` package written in `S-Plus`. The package is available from Statlib *(http://lib.stat.cmu.edu/S/)*. If we attempt to use this procedure with 100,000 points with a small grid of 10×10, the amount of memory required is astronomical.

Sub Sampling

One approach is to use a subsample of the data. We can draw a sample of 5000 data points, which is the largest that the `kde2D` density estimation algorithm can handle, and use these to form the density estimate. The procedure can be repeated for another sample of 5000 points and the two densities can be compared. If the estimates are different then this will tell us that we are subject to sampling error and need more points.

We know that in order to get good convergence of the Gibbs Sampler, we would like to run the process for as long as is computationally possible. By subsampling, we are throwing away a lot of these valuable simulations.

Results

We subsampled from the 100,000 data points and used a bivariate normal kernel with bandwidth 0.3, chosen from the data by the package, to get estimates of the surface. The results for subsamples of 500, 1000, 2000 and 5000 points are shown in Figure 4-8.

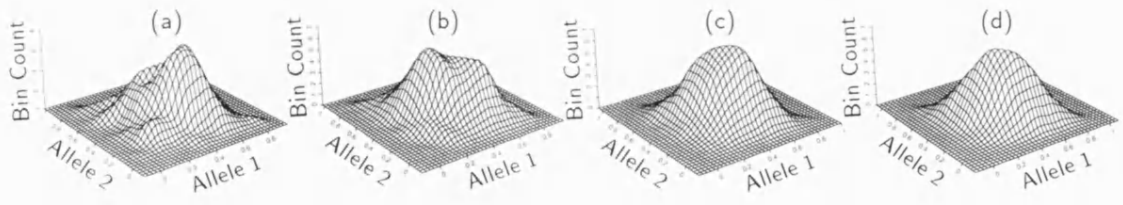


Figure 4-8: Subsampled Gaussian Kernel Surface Estimates: The four perspective plots show the density surface obtained by drawing a subsample from the 100,000 realisations from the Bayes Gibbs algorithm for trait C71. The size of the subsample is 500, 1000, 2000 and 5000 points respectively. For each subsample size a 2D Gaussian kernel density estimate is performed on a grid of 40 by 40 anchor points with kernel width 0.3.

Only one diagram for each subsample is shown but in practice we repeated the process to see if subsampling made any difference to the shape of the density estimate. With only 500 samples the estimate varies dramatically for each subsample. As the number of realisations in the sample increases the process does begin to show the unimodal surface that we are looking for. The process also becomes less erratic but for even 5000 subsamples, differences are still seen in the surface estimate. This implies that the surface is still affected by the sampling process and so more data points are required. This is beyond the current routine for density estimation with its memory eating algorithms.

New Density Estimation Algorithm

We have created a different way of doing 2D Gaussian kernel density estimation. Instead of storing the Gaussian kernel contributions to the anchor points in a matrix – the method used by `kde2D` – we use the increased speed of looping in `S-Plus` and simply sum the contributions at each of the anchor points for all 100,000 data points. This eradicates the need for storing vast amounts of data and the increased looping speed means that this method takes slightly shorter than the `kde2D` package for 5000 points.

Using our new density algorithm we can in theory obtain estimates of the likelihood surfaces from as many realisations as we wish; however in practice, computational run time from the `S-Plus` program does limit the number of realisations. We can now investigate the effect of bandwidth.

Bandwidths

The results for four bandwidths – 0.01, 0.05, 0.1 and 0.2 – of the 2D Gaussian kernel using a 20×20 grid for the 100,000 realisations are shown in Figure 4-9.

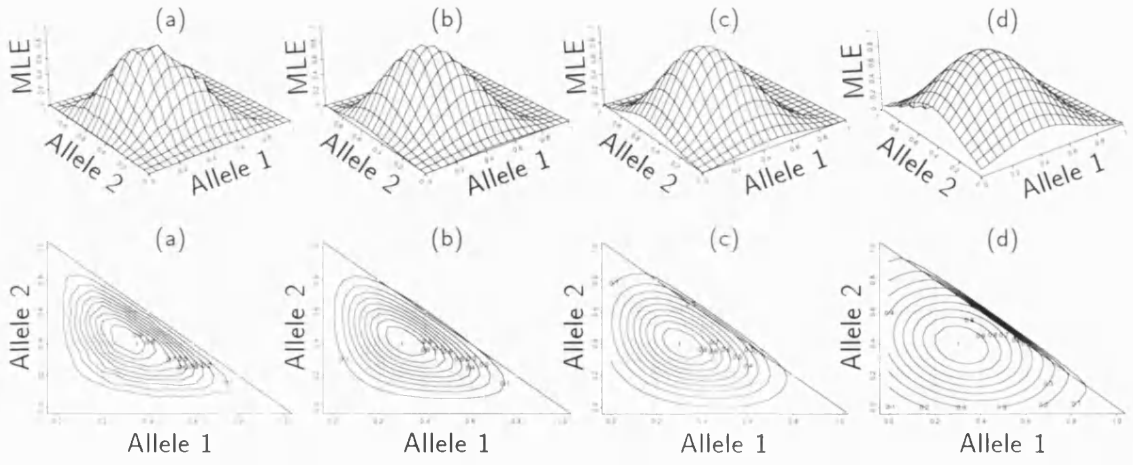


Figure 4-9: Effect of Bandwidth on Gaussian Kernel Estimate: The four perspective and contour plots show the density surface obtained from the 100,000 realisations from the Bayes Gibbs algorithm for trait C71. For the sample of 100,000 realisations a 2D Gaussian kernel density estimate is performed on a grid of 20 by 20 anchor points with four kernel bandwidths: 0.01, 0.05, 0.1 and 0.2.

The estimates of the surface using the Gaussian kernel appear to be smoother than the estimates obtained from interpolating the histograms. The estimates also show the effect of the bandwidth. Increasing the bandwidth from 0.01 to 0.2 changes the estimate of the surface from one which is undersmoothed to one which is oversmoothed. As we have the luxury of comparing these estimates with the exact surface shown in Figure 2-3 we can see that a width of between 0.05 and 0.1 gives the best estimate of the surface.

It is interesting to note that apart from the smallest width, the position of the MLE, denoted by a 1 in the contour plots, hardly moves for different kernel bandwidths and

appears to give an MLE $\approx$ (0.28, 0.40). This is not too close to the exact MLE of (0.2687,0.4510).

Grid Points

Can the estimate be improved by increasing the number of grid points at which the Gaussian kernel estimate is calculated? The results for increasing numbers of grid points with constant bandwidth of 0.05 are shown in Figure 4-10.

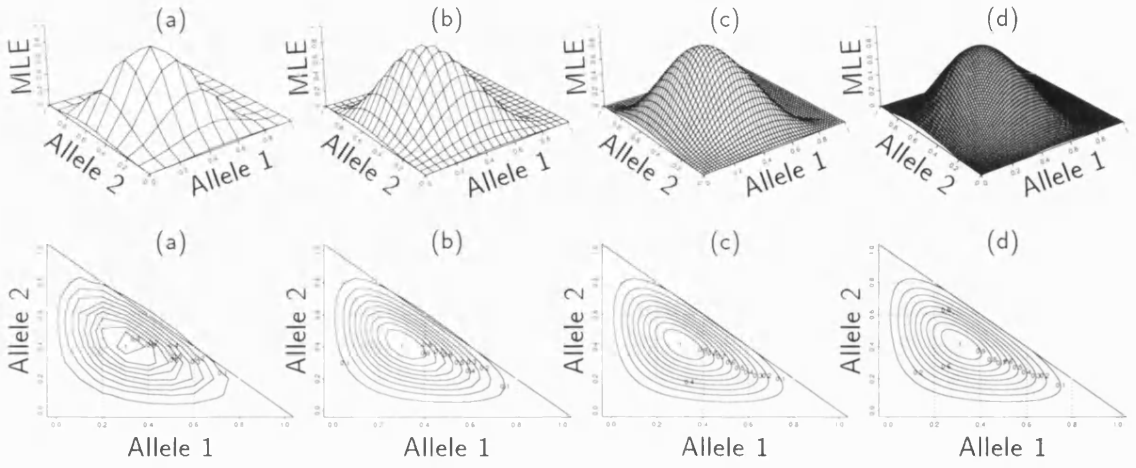


Figure 4-10: Effect of Grid Points on Gaussian Kernel Estimate: The four perspective plots show the density surface obtained from the 100,000 realisations from the Bayes Gibbs algorithm for trait C71. For the sample of 100,000 realisations a 2D Gaussian kernel density estimate is performed with bandwidth 0.05 on four different grid sizes: 10, 20, 50 and 100.

For a grid of 10×10 the contour plots are jagged and more anchor points are required. Increasing the grid of points appears to make little difference to the surface estimate although the MLE moves very slightly.

Conclusions

All the results in this section on density estimation were based on a sample of 100,000 realisations of the Bayes Gibbs algorithm for trait C71.

The conclusions seem to be that with so many points it makes very little difference what kind of density estimation you perform. The Gaussian kernel method produces slightly better and smoother surfaces than the interpolation of the histogram method. The estimate of the likelihood surface is very good when compared with the exact surface shown in Figure 2-3; however, the results for the simulated MLEs are not that close to the exact MLEs. This is understandable as on top of the relaxation parameter in the Bayes Gibbs algorithm, we also have to adjust the parameters of the density estimation process: bandwidth, number of histogram bins, and grid points.

In order to increase accuracy of the process we need to collect more realisations and this is going to put computational restrictions on the Gaussian kernel method. So, as we increase the number of realisations we will use a simple binning routine to create a 2D histogram, on which will use our interpolation routine to create a smooth surface estimate.

4.4.5 Increasing Simulations

In this section we investigate the effect of increasing the number of realisations on the estimate of the surface. For the two allele traits, we used 100,000 draws, 40 histogram bins and succeeded in getting a very good estimate of the likelihood curves. For the three allele trait C71, we have found that a grid of 20×20 histogram bins scaled up to 40×40 using the interpolation routine to get a smooth surface estimate worked well. Due to the unity condition, this means that we are considering 200 bins which gives fewer realisations per bin and is going to lead to a poorer estimate of the likelihood surface. To try and overcome this, we try to increase the number of realisations of the Bayes Gibbs algorithm for trait C71.

As we have previously mentioned, it is computationally cheap to generate realisations from the likelihood surface using the Gibbs Sampler. Clearly, the more draws we achieve the better the estimate of the MLEs as it will not be so susceptible to the number of bins. However, a statistical package such as S-Plus can only handle about 250,000 draws of a three allele trait before it runs out of memory. So in order to get large number of draws we have to bin these 250,000 draws and keep a table of the bin counts. The whole process can be repeated and all that the package has to do is add two tables of bin counts together and restore the new table of counts. This allows us to do as many draws using the Bayes Gibbs algorithm as we wish.

We run the Bayes Gibbs algorithm on trait C71 for two million realisations using a γ of 0.01 which takes approximately 20 hours. Figure 4-11 shows the estimates of the likelihood surface after one and two million realisations. The surfaces were obtained by using a grid of 20×20 bins which is then increased by the interpolation routine to 40×40 grid points.

Results

The contour plots of the estimate of the surface show that the increased number of realisations has improved the estimate from Figure 4-7(b). The MLE, denoted by a 1 on the plots, is more robust and is nearer to the exact value of (0.2687, 0.4510). However, the surface estimate is not as smooth as the estimates obtained from using Gaussian kernels. We remember that we have increased the new number of realisations by factors of 10 and 20 respectively and in order to use this method on the four, five and six allele traits, the number of realisations from the Bayes Gibbs algorithm is going to be huge, even if we use a relatively small number of bins.

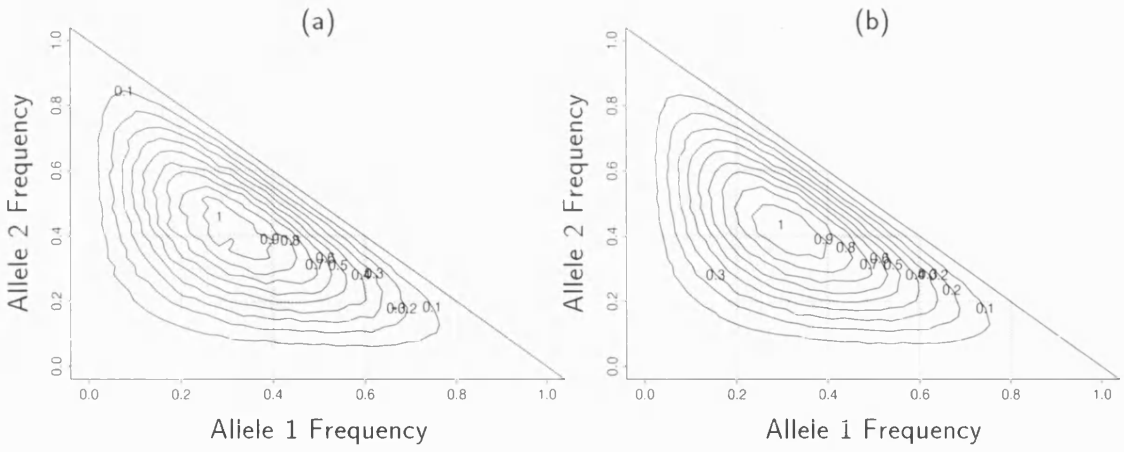


Figure 4-11: EFFECT OF INCREASED REALISATIONS FOR TRAIT C71: The estimates of the likelihood surface using one and two million realisations for trait C71 are shown. The surfaces were obtained by initially using a grid of 20×20 bins which is then increased by the interpolation routine to a grid of 40×40.

4.4.6 Conclusions

The algorithm that we have used in this section shows that it simulates from the correct posterior distribution and, as we have chosen a constant prior, consequently approximates the exact likelihood surfaces very well. However, we are having problems getting an estimate of the MLEs to any degree of accuracy. In order to get good approximations, we need to run the algorithm for a considerable number of realisations and this makes the choice of which density estimation procedure to use unimportant. The Gaussian kernel estimate does produce smoother estimates of the likelihood surface than the interpolated 2D histogram routine but also adds substantial computational cost without providing a dramatic improvement. So for simplicity, we have used a simple 2D histogram but have found that increasing the dimension from two alleles to three alleles means that we have to increase the number of realisations to maintain the accuracy of the MLEs. This will continue if we look at the higher allele traits and we have the problem of trying to quantify our mode in 5D space.

So at this point we decide to bail out of trying to find MLEs using this method, as in the previous chapter we developed a Gibbs EM algorithm which provides us with the MLEs. We consider the value of the new Bayes Gibbs algorithm is in its ability to produce a good estimate of the posterior or likelihood distribution. The easiest method of checking the accuracy of the Bayes Gibbs process is to compare the estimated marginal means with the exact marginal means. So we now consider the ability of the algorithm to converge to the marginal means.

4.5 Marginal Means

4.5.1 The Three Allele Trait - C71

In §4.3 we found that the Bayes Gibbs algorithm worked very well for the marginal means for the two allele traits. Results for 100,000 draws of the algorithm were very good. We found that no increase in accuracy was obtained running more than one Gibbs sweep per prior update and as computational time must be a consideration, it was better to run the process for longer.

In this section we run the process on trait C71 for several million draws to give an idea of how long we have to run to obtain convergence for a predetermined degree of accuracy. The Bayes Gibbs algorithm is computationally cheap and so we can build up these millions of samples quickly. In §4.4.1 we showed that increased correlation meant we should run the algorithm with as high a γ as computational cost could allow. For these long runs, we use γ to 0.005. Under this regime, one million realisations takes approximately three hours and yields a rejection rate of approximately 65%. A problem occurs as a data manipulation package such as `S-Plus` can only handle about 250,000 numbers at a time. To avoid this we keep records of the $\sum x_i$ and $\sum x_i^2$ to allow us to calculate the mean and standard deviation. Table 4.5 gives the results for the marginal means and their associated deviation estimate for an increasing number of realisations up to five million. The exact $E(X_i)$s are shown for comparison and are obtained from the exact likelihood surface by the method outlined in Equation (4.13). Hence for a three allele trait, where $X_1 + X_2 + X_3 = 1$, $E(X_1)$ when $X \sim c\, p(x)p(\phi(D) \mid X = x)$ is

$$E(X_1) = c \int_0^1 \int_0^1 p(x)p(\phi(D) \mid X = x)\, x_1 \;\; x_2 \, dx_2 \, dx_1 \tag{4.19}$$

Results

The results are good. After only 100,000 realisations the estimates are very accurate and very little accuracy is gained in running the process for longer than approximately one million realisations.

The Bayes Gibbs process produces estimates of the standard deviations as a matter of course. Again, the standard deviations appear to have settled down by approximately one million realisations and in fact, rather accurate standard deviations are available from only 100,000 realisations.

Effective Binomial Sample Size

Using the estimates of the marginal means and standard deviations we can obtain estimates of the effective *Binomial* sample size. The idea is that the marginal posterior distribution for a given allele frequency should be roughly equivalent in information content to directly observing n *Bernoulli* outcomes on that allele with frequency θ, where n

Point Estimates of the Allele Frequencies for an Increasing Number of Realisations				
Nos of Realisations		Allele 1	Allele 2	Allele 3
100,000	Mean	0.3233	0.3895	0.2875
	SD	0.1554	0.1597	0.1410
200,000	Mean	0.3250	0.3889	0.2861
	SD	0.1557	0.1604	0.1407
500,000	Mean	0.3239	0.3895	0.2866
	SD	0.1551	0.1610	0.1412
1,000,000	Mean	0.3230	0.3902	0.2867
	SD	0.1548	0.1606	0.1411
2,000,000	Mean	0.3235	0.3897	0.2868
	SD	0.1550	0.1608	0.1415
3,000,000	Mean	0.3235	0.3898	0.2868
	SD	0.1549	0.1609	0.1415
4,000,000	Mean	0.3237	0.3898	0.2866
	SD	0.1550	0.1608	0.1414
5,000,000	Mean	0.3238	0.3897	0.2865
	SD	0.1549	0.1609	0.1415
Exact Mean		0.3229	0.3908	0.2863
Effective *Binomial* Sample Size		9.0	9.3	10.3

Table 4.5: Point Estimates for Trait C71: The table displays the estimated marginal means and standard deviations for the founding allele frequencies for trait C71. The estimates are derived from an increasing number of realisations of the Bayes Gibbs sampling process. The exact means obtained from the peeling method are shown for comparison. Also shown is the effective *Binomial* sample size.

may be determined at least approximately by equating θ to the posterior mean and then solving

$$\text{Posterior SD} = \sqrt{\frac{\theta(1-\theta)}{n}}. \tag{4.20}$$

for n.

The results for each of the alleles are also shown in Table 4.5 and they indicate that for trait C71 the data information is the same as observing the truth on about 9–10 horses.

Beta Comparison

In the C71 trait all the effective *Binomial* sample sizes based on the simple idea above are approximately 10 and the marginal posterior mean estimates are not far off parity. However, for the four allele trait below, we find one of the effective *Binomial* sample sizes is much larger than the other three; it corresponds to a marginal posterior mean estimate which is very small. So there is a worry that Equation (4.20) might be inaccurate for highly skewed θ. We can check the accuracy of the effective *Binomial* sample sizes by using the conjugacy of the *Binomial* distribution with the *Beta* distribution. In order to

get estimates of the *Binomial* sample sizes we have to find the *Beta* distribution that best fits the marginal distributions.

The method that we have chosen avoids the MLE approach outlined in Johnson and Kotz (1970) and uses a quick and simple approach.

Algorithm 4.4. BETA FITTING:

1. *Generate some realisations from the Bayes Gibbs algorithm.*
2. *For each of the alleles bin the realisations into a histogram and standardise.*
3. *At the midpoints of the histogram bars calculate the points of a* Beta *distribution and sum the absolute differences between the histogram points and the* Beta *points.*
4. *Repeat step 3 for different parameters of the* Beta *distribution until the difference is minimised.*

□

We note that this method equally weights all discrepancies and that due to an increase in dimensions, it would be possible to fit a *Dirichlet* distribution rather than fitting three Beta distributions.

However, we used the *Beta* Fitting algorithm to find the *Beta* distributions that best fit the marginal distributions for 100,000 realisations of the Bayes Gibbs algorithm for trait C71. The parameters of the *Beta* distribution used are on a grid between 0–10 in steps of 0.1.

Results

The results of the best fitting *Beta* distribution for the three marginals using the above regime and the simulated realisations are displayed in Figure 4-12. The plots are formed by linearly joining the estimated and actual histogram bar centres. This is done to show up discrepancies in the two distributions.

For trait we find that a *Beta*(2.6,5.4) for allele 1, a *Beta*(3.2,5.0) for allele 2 and a *Beta*(2.7,6.7) for allele 3 best fit the simulated data. Having found these parameters we can now estimate the *Binomial* sample size. It is as if the $Beta(\alpha, \beta)$ distribution was worth $(\alpha + \beta)$ *IID Bernoulli* observations and so for the respective alleles this equates to effective *Binomial* sample sizes of 8.0, 8.2 and 9.4 which are close to the estimates given in Table 4.5 obtained through Equation (4.20).

As we have the parameters of the *Beta* distribution that best fits our realisations from the marginal distributions, we can also check some of the other estimates in the table such as the marginal means and standard deviations. Table 4.6 compares the estimates of the marginal means and their associated standard deviations obtained from the Bayes Gibbs process with the estimates obtained from the *Beta* fitting algorithm.

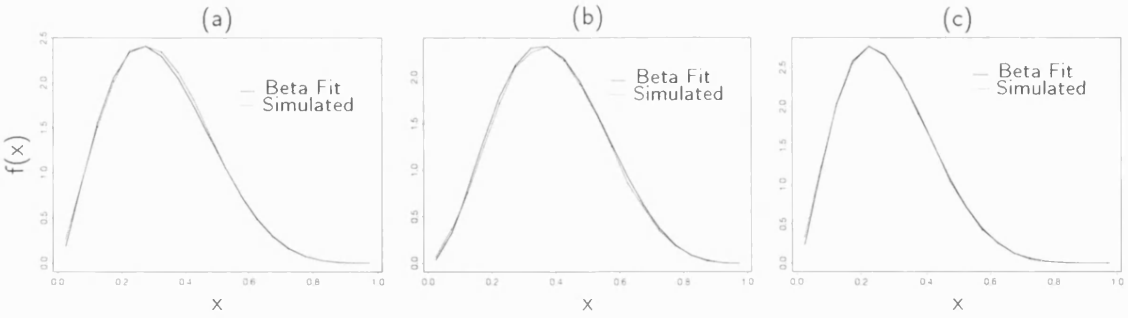


Figure 4-12: Marginal Plots of Bayes Gibbs Algorithm for Trait C71: The figure plots a comparison of the simulated marginal distribution of all three alleles for trait C71. In each case the simulated marginal distribution is plotted against the Beta distribution which best fits it. Plots (a)–(c) are for alleles 1–3 respectively.

Comparison of Marginal Means for Trait C71			
	Allele 1	Allele 2	Allele 3
Bayes Gibbs Mean	0.3238	0.3897	0.2865
Bayes Gibbs SD	0.1549	0.1609	0.1415
Effective *Binomial* Sample Size	9.0	9.3	10.3
Beta Fitted Parameters	2.6, 5.4	3.2, 5.0	2.7, 6.7
Beta Fitting Mean	0.3250	0.3902	0.2872
Beta Fitting SD	0.1561	0.1608	0.1403
Effective *Binomial* Sample Size	8.0	8.2	9.4
Exact Mean	0.3229	0.3908	0.2863

Table 4.6: Comparison of Marginal Means for Trait C71: The table compares estimates of the marginal means and their associated standard deviations obtained from the Bayes Gibbs algorithm and the *Beta* Fitting algorithm. The exact marginal means are shown for comparison. Also shown is the effective *Binomial* sample size obtained from the *Binomial* approximation and the *Beta* fitting algorithm.

From the table we can see that the *Binomial* sample sizes using both the *Binomial* approximation and *Beta* fitting are close. The estimates of the marginal means are approximately equally close to the exact marginal means for both methods.

Results for the other two three allele traits, C72 and Pa, show similar results to those for C71. The results are not shown to save the reader wading through pages of similar analysis. Estimates of the marginal means rapidly achieve accuracy to three decimal places and the effective *Binomial* sample size is approximately 10 for all alleles.

4.5.2 The Four Allele Trait

We now use the Bayes Gibbs algorithm on the four allele trait Serum Es. As in previous sections on these higher allele traits, we cannot check the simulated results with any exact calculations. However, we can run the process with different seeds for the random number generator and look at the statistics to see if different values are achieved.

We run the process re-seeding the random number generator after every 50,000 draws from the sampling process. As in §3.6, the section on the Gibbs EM algorithm for this trait, the relaxation parameter used is reduced to 0.0005 and like previous simulations on this trait we find that it takes the longest amount of run time with one million realisations taking approximately 20 hours and a rejection rate of approximately 90%. The results for the marginal means and their associated standard deviations for increasing numbers of realisations of the Bayes Gibbs algorithm using this regime are given in Table 4.7.

Point Estimates of the Allele Frequencies for Trait Serum Es					
Nos of Realisations		Allele 1	Allele 2	Allele 3	Allele 4
50,000	Mean	0.1977	0.2289	0.4945	0.0789
	SD	0.0991	0.1056	0.1239	0.0533
100,000	Mean	0.1978	0.2297	0.4936	0.0789
	SD	0.0995	0.1053	0.1237	0.0532
250,000	Mean	0.1978	0.2307	0.4927	0.0788
	SD	0.0992	0.1055	0.1238	0.0532
500,000	Mean	0.1979	0.2311	0.4924	0.0786
	SD	0.0995	0.1055	0.1242	0.0533
750,000	Mean	0.1982	0.2306	0.4925	0.0787
	SD	0.0995	0.1055	0.1241	0.0530
1,000,000	Mean	0.1982	0.2307	0.4925	0.0787
	SD	0.0996	0.1055	0.1241	0.0530
Effective *Binomial* Sample Size		16.0	15.9	16.2	25.8

Table 4.7: Point Estimates for Trait Serum Es: The table displays the estimated marginal means and their associated standard deviations for the founding allele frequencies for trait Serum Es. The estimates are derived from an increasing number of realisations of the Bayes Gibbs sampling process. Also shown is the effective *Binomial* sample size obtained through the *Binomial* approximation.

Results

As we have no exact answers to compare with we use convergence as a guide. Examining each re-seeded set of 50,000 realisations shows only small differences in the marginal mean and standard deviation statistics. The algorithm seems to converge quickly. with all estimates being different by less than 0.002 after 50,000 realisations compared to one million realisations.

The estimates of the effective *Binomial* sample size are not the same for the Serum Es marginal distributions with the result for allele 4 being substantially larger than the remaining three alleles. We double check these results using our *Beta* fitting algorithm.

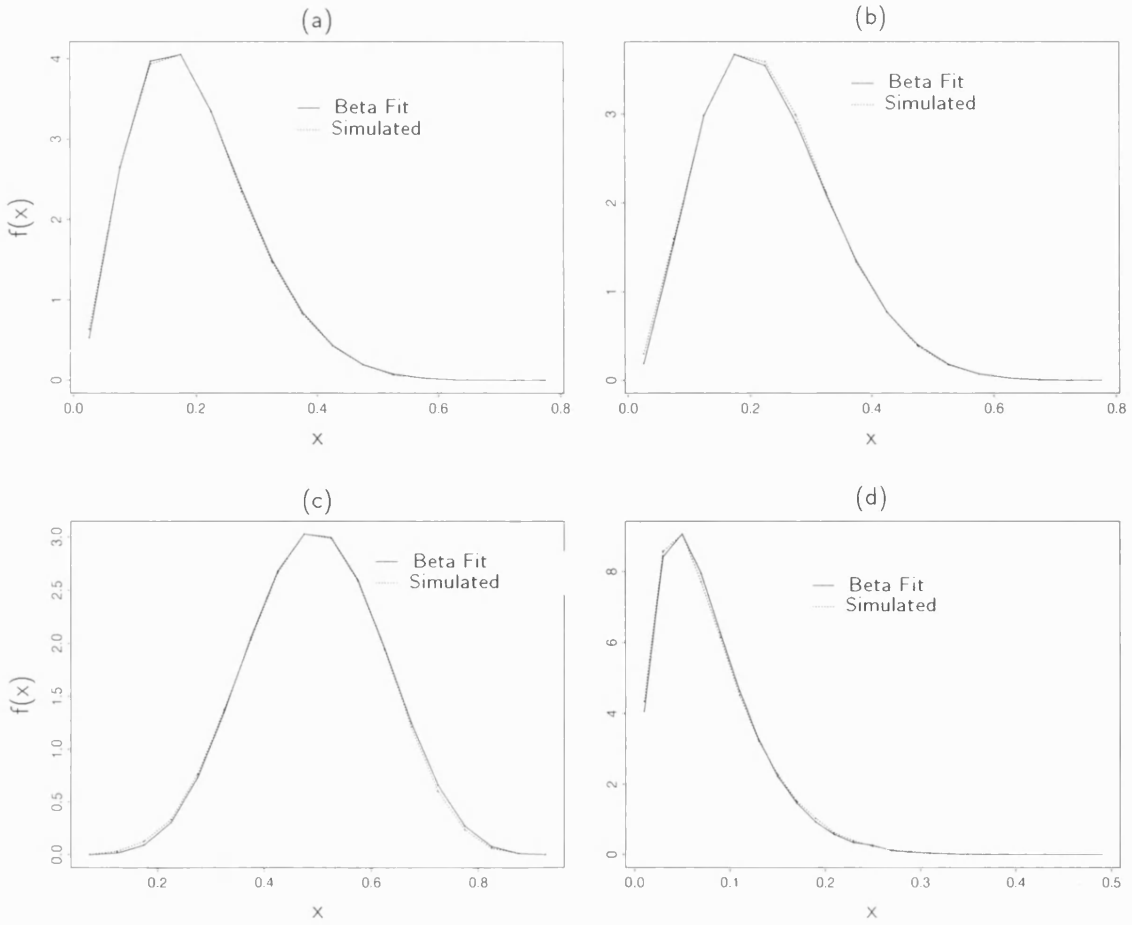


Figure 4-13: Marginal Plots of Bayes Gibbs Algorithm for Trait Serum Es: The figure plots a comparison of the simulated marginal distribution of all four alleles for trait Serum Es. In each case the simulated marginal distribution is plotted against the Beta distribution which best fits it. Plots (a)–(d) are for alleles 1–4 respectively.

Beta Comparison

The plots of the estimated marginal distributions and the best fitting *Beta* distribution are shown in Figure 4-13. The plots again use 100,000 realisations but this time in order to obtain a *Beta* distribution that fits the simulated data satisfactorily, the grid of parameters that we use to run the fitting algorithm is extended to run between 0–30 for the parameters of the *Beta* distribution. The plots again use discrete lattices connected with line segments to show up any discrepancies.

Results

For trait Serum Es we find that a $Beta(3,12.1)$ for allele 1, $Beta(3.4,11.4)$ for allele 2 $Beta(7.5,7.6)$ for allele 3 and $Beta(2.1,24.5)$ for allele 4 best fit the simulated marginal distributions.

Again as we have the parameters of the *Beta* distribution that best fits the simulated data, we can obtain estimates of the marginal mean and its standard deviation as well as the effective *Binomial* sample size. The results are shown in Table 4.8.

Comparison of Marginal Means for Trait Serum Es				
	Allele 1	Allele 2	Allele 3	Allele 4
Bayes Gibbs Mean	0.1982	0.2307	0.4925	0.0787
Bayes Gibbs SD	0.0996	0.1055	0.1241	0.0530
Effective *Binomial* Sample Size	16.0	15.9	16.2	25.8
Beta Fitting Parameters	3, 12.1	3.4, 11.4	7.5, 7.6	2.1, 24.5
Beta Fitting Mean	0.1987	0.2297	0.4967	0.0789
Beta Fitting SD	0.0994	0.1058	0.1246	0.0513
Effective *Binomial* Sample Size	15.1	14.8	15.1	26.6

Table 4.8: Comparison of Marginal Means for Trait Serum Es: The table compares estimates of the marginal means and their associated standard deviation obtained from the Bayes Gibbs algorithm and the *Beta* Fitting algorithm. Also shown is the effective *Binomial* sample size obtained from the *Binomial* approximation and the *Beta* fitting algorithm.

For trait Serum Es the results for both the *Binomial* approximation and the Beta fitting are shown in Table 4.8. The values of the effective *Binomial* sample size using both the *Binomial* approximation and the *Beta* fitting algorithm are consistent and show that allele 4 does have a larger effective sample size than the other three alleles. The estimates of the marginal means and their associated standard deviations are again similar but the *Beta* Fitting estimates appear to be closer to the Bayes Gibbs estimates than for trait C71; this is due to running the *Beta* Fitting algorithm over a larger range of parameter values and consequently obtaining a better fit to the simulated data.

4.5.3 The Five Allele Trait

We now use the Bayes Gibbs algorithm on the five allele trait Pr to give us an estimate of the marginal means and standard deviations.

We run the process re-seeding the random number generator after every 25,000 draws from the sampling process. As in §3.7, the section on the Gibbs EM algorithm for this trait, we reduce γ to 0.0005. The run time for one million realisations is approximately seven hours with a rejection rate of approximately 50%. The results for the marginal means and their associated standard deviations for groups of realisations of the algorithm up to one million using this regime are given in Table 4.9.

Results

The results again show that the algorithm settles down very quickly and although there is a suggestion that it takes a little longer to converge than for the three and four allele traits, convergence appears to have been reached by one million realisations.

Point Estimates of the Allele Frequencies for Trait Pr						
Numbers of Realisations		Allele				
		1	2	3	4	5
50,000	Mean	0.4975	0.2110	0.1154	0.0877	0.0884
	SD	0.1471	0.1107	0.0801	0.0668	0.0671
100,000	Mean	0.4976	0.2102	0.1166	0.0875	0.0883
	SD	0.1462	0.1101	0.0806	0.0669	0.0669
250,000	Mean	0.4992	0.2107	0.1163	0.0868	0.0870
	SD	0.1465	0.1103	0.0805	0.0663	0.0662
500,000	Mean	0.4997	0.2109	0.1164	0.0865	0.0865
	SD	0.1463	0.1103	0.0805	0.0660	0.0659
750,000	Mean	0.4994	0.2111	0.1163	0.0866	0.0864
	SD	0.1464	0.1106	0.0806	0.0660	0.0659
1,000,000	Mean	0.4993	0.2113	0.1162	0.0866	0.0865
	SD	0.1463	0.1106	0.0806	0.0661	0.0659
Effective *Binomial* Sample Size		11.7	13.6	15.8	18.1	18.2

Table 4.9: Point Estimates for Trait Pr: The table displays the estimated marginal means and their associated standard deviations for the founding allele frequencies for trait Pr. The estimates are derived from an increasing number of realisations of the Bayes Gibbs sampling process. Also shown is the effective *Binomial* sample size.

4.5.4 The Six Allele Trait

We now use the Bayes Gibbs algorithm on the six allele trait Tf to give us an estimate of the marginal means and associated standard deviations.

We run the process re-seeding the random number generator after every 25,000 draws from the sampling process. Again as in §3.8, the section on the Gibbs EM algorithm for this trait, we reduce γ to 0.05%. The run time for one million realisations is approximately 10 hours with a rejection rate of approximately 60%. The results for the marginal means and their associated standard deviations for groups of realisations of the algorithm up to one million using this regime are given in Table 4.10.

Results

Again convergence is achieved but there is even more of a suggestion that more than one million realisations are needed for convergence to four decimal place accuracy especially with the marginal mean for allele 6 which is still changing.

4.6 Conclusions

In this chapter we have introduced and tested a method called the Bayes Gibbs algorithm. The algorithm uses a constant *Dirichlet* prior and incorporates the data in a single Gibbs sweep. The result is a realisation from the full posterior distribution which we have set

Point Estimates of the Allele Frequencies for Trait Tf							
Numbers of Realisations		Allele					
		1	2	3	4	5	6
50,000	Mean	0.3090	0.1677	0.1855	0.1152	0.0883	0.1345
	SD	0.1402	0.1048	0.1089	0.0778	0.0640	0.0081
100,000	Mean	0.3119	0.1678	0.1833	0.1146	0.0882	0.1344
	SD	0.1392	0.1051	0.1079	0.0776	0.0641	0.0809
250,000	Mean	0.3165	0.1690	0.1791	0.1131	0.0877	0.1347
	SD	0.1403	0.1051	0.1066	0.0772	0.0638	0.0813
500,000	Mean	0.3186	0.1686	0.1783	0.1129	0.0875	0.1342
	SD	0.1405	0.1052	0.1062	0.0773	0.0636	0.0809
750,000	Mean	0.3173	0.1692	0.1784	0.1132	0.0876	0.1349
	SD	0.1405	0.1054	0.1064	0.0775	0.0637	0.0808
1,000,000	Mean	0.3171	0.1693	0.1783	0.1132	0.0877	0.1344
	SD	0.1404	0.1054	0.1064	0.0774	0.0637	0.0809
Effective *Binomial* Sample Size		11.0	12.7	12.9	16.8	19.7	17.8

Table 4.10: Point Estimates for Trait Tf: The table displays the estimated marginal means and their associated standard deviations for the founding allele frequencies for trait Tf. The estimates are derived from an increasing number of realisations of the Bayes Gibbs sampling process. Also shown is the effective *Binomial* sample size.

up using a constant prior to be proportional to the likelihood. By running some form of density estimation on the realisations we can easily get an estimate of the likelihood and we have found that the estimate when compared to the exact likelihood in Figure 2-3, is good.

In this thesis we have been mainly involved in trying to find the MLEs – or marginal posterior modes – of the founding population. We have found that whichever density approach we use, despite getting a good approximation of the likelihood, the MLE is highly subject to the parameters of the density estimation procedure. In the cases of the two and three allele traits, where the exact MLEs are known, we can adjust these parameters to get good estimates for the MLEs from the Bayes Gibbs realisations but for the higher allele traits, the MLEs are what we want to estimate and so the process is highly sensitive to parameter specification. In any case, in Chapter 3, we introduced the Gibbs EM algorithm which find the MLEs very accurately and efficiently.

The advantage for the Bayes Gibbs algorithm is in obtaining a good estimate of the full likelihood or posterior distribution and in the estimation of the marginal means and associated statistics, which have not been of principal interest in this thesis but may be of interest in other projects. The Bayes Gibbs algorithm does estimate the marginal means extremely rapidly and efficiently for all of the traits. These estimates, both for the marginal means and their standard deviations, have been checked by fitting a *Beta* distribution to a histogram of some Bayes Gibbs realisations and obtaining estimates from the *Beta* parameters.

Chapter 5

Conclusions and Further Work

You've got to accentuate the positive,
Eliminate the negative,
Latch on to the affirmative,
Don't mess with Mr In-between.
(Johnny Mercer)

The investigation of MCMC methods in pedigree analysis in this thesis was started by a biologist who asked a simple question. By looking at the founders of the pedigree, could we provide any information on whether two different management programs, currently in operation for the pedigree of the Przewalski horse, were justified? Phenotypic information was available for approximately 60% of the horses on sixteen different traits ranging from two to six alleles. Exact results using an existing algorithm, peeling, could be obtained for the lower allele traits but simulation methods needed to be investigated for the higher allele traits.

We have developed three methods for finding the MLEs of the founding population which are needed for accurate ancestral inference and so for answering the initial question of interest.

- Initially, we nest the peeling algorithm within an EM structure.

- To achieve answers for the higher allele traits we replace the peeling algorithm in the EM structure with the simulation algorithm, the Gibbs Sampler.

- To try and improve the technique we adopt a fully Bayesian framework. By using a constant *Dirichlet* prior, the likelihood is then proportional to the posterior distribution. We then use the Gibbs Sampler to sample from this distribution.

A chapter breakdown of conclusions and future work is now stated.

Chapter 1

Conclusions

- From the brief review of pedigree analysis in Chapter 1, it is clear that the handling of rare and endangered species is going to be increasingly important. Consequently, mathematical methods that will provide answers to management questions will be in great demand. Any method that works and helps pedigree managers in making decisions on breeding policy will be of importance.

Further Work

- To continue answering management questions for this pedigree which is being used as a blueprint for further endangered captive pedigrees.

Chapter 2

Conclusions

- The peeling EM algorithm, developed to find the MLEs of the allele frequencies of the founders, works well and converges quickly. The EM structure does rely on an assumption of unimodality to converge to the MLEs and so time has to be spent checking for multi-modality by starting the algorithm at different initial points around the sample space. Once we have satisfied ourselves that we have a unimodal surface, the algorithm can be run for longer and can rapidly achieve extremely accurate results. However, the peeling EM is limited by the vast memory requirements that it requires. We do what we can to reduce the memory requirements by clipping the pedigree and using a stochastic relaxation algorithm – simulated annealing – to reorder the peeling summations, but only results on the two and three allele traits of the pedigree can be calculated.

- Having used the peeling EM algorithm to find the MLEs of the founding population, we then performed the ancestral inference on the six founding groups to answer the questions about the management programs. We found a huge amount of variety in the allele frequencies of the founding groups. This suggests a large amount of interbreeding between domestic and Przewalski horse on the Mongolian plains and makes the management programs. which are based on 'pure' and 'contaminated' horses, difficult to justify.

- There is no way that the peeling algorithm can give us error estimates and we resort to fitting a quadratic spline around the MLEs to obtain uncertainty estimates.

Further Work

- We could speed up the convergence of the peeling EM algorithm by using some kind of projection method from the size of the jumps of the EM iterations.

Chapter 3

Conclusions

- The Gibbs EM algorithm works very well when compared to the exact results from the peeling EM algorithm. It is quicker and in this respect, outperforms the peeling EM algorithm in Chapter 2. However, we have to check the algorithm and its run parameters more closely. Most important is that that for multi-allele traits, irreducibility of the Markov Chain is not guaranteed due to the laws of Mendelian inheritance. We get around this by using a relaxation algorithm. This gives a small positive probability of obtaining all pedigree configurations, and hence irreducibility is now achieved. However, configurations are generated in the sampling process that break the laws of inheritance; these are rejected. The relaxation technique for this pedigree is found to work well and the associated relaxation parameter γ of 0.005 appears to give the best compromise between computational cost, serial correlation and accuracy. Again time has to be spent checking for multi-modality of the likelihood surface. This can use a small number of Gibbs samples. We have found that 10 iterations of 100 sweeps per iteration is enough to suggest convergence for all the multi-allele traits considered; so it is computationally very quick but increased repetition of the process from different starting points around the sample space increases our confidence that any convergence is to the MLEs. In this pedigree we are fortunate and have found no trait that shows a multi-modal surface.

- There are problems visualising convergence of the Gibbs EM algorithm in higher dimensions. We have used multi-panel two allele displays which we have found best.

- Having used the short runs to tune the run parameters of the algorithm, we can use a longer run of the process to obtain estimates of the MLEs of the founders' allele frequencies. After a 5% burn-in period, the algorithm is usually so close to the MLE that very few iterations are needed. Typically 100 EM iterations of 10,000 sweeps per iteration gives accuracy to three decimal places. However, it is important to remember that for the higher allele traits, for which we have no exact answers to check against, monitoring the convergence of the estimates is probably the best method rather than using a certain regime. For example, although all the traits that we have looked at converge relatively quickly, there is a suspicion that the six allele trait does take longer to converge to four decimal place accuracy than the other traits.

- Having investigated the Gibbs Sampler and shown that it can produce good probability estimates on the pedigree, we performed the analysis for the three multi-allele on the six founding groups of the pedigree to provide further evidence on the relevance of the management programs. We found similar results to the two and three allele traits. The founding groups show a large genetic diversity indicative of a large amount of interbreeding in the ancestral population. This again makes the management programs with their additional mild form of selection difficult to justify. Further evidence is provided when we looked at the founding and current population allele frequency estimates. Since we have more alleles in these traits than in the two and three allele traits, some of the alleles are quite rare in the founding population and because of management pressure, the rarer alleles have dropped to an extent where they are in danger of disappearing from the current population altogether.

Further Work

- We have found that for this pedigree a simple Gibbs Sampler gives good results. However, alternative methods which deal with the problems of multi-modality and faster mixing, such as Heated–Metropolis and simulated tempering should be investigated. Some of these alternative samplers are outlined in §5.1.1 and §5.1.2.

- A new algorithm, circumspectly entitled the *Random Family* algorithm by Hurn and Sheehan (personal communication), based on the relaxation technique of Sheehan and Thomas (1993), should be implemented and investigated. Details of this algorithm are given in §5.1.4.

Chapter 4

Conclusions

- The Bayes Gibbs algorithm samples from a constant *Dirichlet* prior distribution and sweeps through the data with a Gibbs Sampler. It works very well and efficiently. Using only one sweep of the Gibbs Sampler per prior update has been found to provide the best results.

- In the numerical work in this dissertation, estimates of the marginal posterior densities have been based on kernel density estimation applied to the output of the Gibbs sampler. When the quantity of interest is a component, say X_1, of the vector parameter, $\pi_F = (X_1, \ldots, X_k)$, an alternative which is theoretically superior involves averaging the full conditional distribution $p(X_1 \mid X_2, \ldots, X_k, \phi(D))$ over the sampled values $(X_2^\star, \ldots, X_k^\star)$. This idea, which was suggested by Gelfand and Smith (1990) based on an appeal to the Rao-Blackwell theorem, may more easily be motivated, as noted by Tanner (1993), by considering Monte Carlo approximation

of the integral

$$p(X_1 \mid \phi(D)) = \int p(X_1 \mid X_2, \ldots, X_k, \phi(D))\, p(X_2^\star, \ldots, X_k^\star \mid \phi(D))\, dx_2 \cdots dx_k.$$

While this method should be better in theory than kernel density estimation in the accuracy of its posterior density summaries, it may be more computationally expensive and would certainly involve greater storage requirements. Quantifying the costs and benefits of this method in the context of the pedigree work in this dissertation is a subject for future investigation.

- Using the Bayesian framework with a constant prior we can use the MCMC method for non-Bayesian likelihood inference. An estimate of the likelihood can be obtained from the MCMC realisations by density estimation. A Gaussian kernel produces a smoother estimate than the interpolated 2D histogram approach but increases the computational demands for only a small improvement.

- There are problems with the density estimation which is required to get an estimate of the MLEs. When comparing the exact and estimated surfaces for the three allele trait C71 we know what the exact surface looks like and can adjust the parameters of the density estimation procedure accordingly to obtain a good estimate. We have found that these parameters do affect the MLEs and finding them accurately in higher dimensions is difficult.

- There is a conflict of wanting more observations from the Markov Chain to estimate the MLEs in higher dimensions and the subsequent strain on the density estimation procedure. We have found that with the large number of realisations only a simple histogram, from which we can use an interpolation routine to get a smooth surface estimate, can be used.

- Again, there are problems visualising the surface estimate in higher dimensions.

- Getting estimates of the marginal means and standard deviations are a consequence of the process and the estimates are very close to the exact values for the three allele trait C71.

Further Work

- The density estimation part of the whole process could be extended to deal only on the simplex. A new approach is outlined in §5.2.1.

- This technique needs to be tried on a larger and more computationally demanding pedigree to see how it performs.

5.1 Alternative Simulation Methods using MCMC

5.1.1 Multi-Modality

It is well known from the ergodic theorem that any Markov Chain simulation model will require a large number of scans in order to achieve convergence to the posterior distribution but a problem occurs since we do not know what the probability density looks like and normally assume it is well behaved and unimodal. However, if the density is multi-modal there exists a high dependency on the initial configuration as to which peak the sampler will converge to. Even worse is the fact that if there is a very low probability for the states in between the modes, one of the modes might never be visited because the sampler would have to run for an incredible length of time in order to pass through these states and swap modes. An example is given in Tavener (1992). He uses a bi-model Ising probability density and shows that if the image is all white pixels, then the probability of changing two chosen pixels to black is of the order 10^{-5}; hence the probability of moving just a small distance away from the top of a mode is very small. Tavener suggests possible remedies in image analysis for cases where the mode positions are known but in pedigree analysis this is not the case and so we have to resort to alternate simulation methods using the Gibbs Sampler.

Sampling from Heated Markov Chains

Jennison (1993) outlines a possible solution to this problem . The author suggests obtaining samples $\mathbf{x}_1 \ldots \mathbf{x}_n$ from the heated distribution

$$P_T(\mathbf{X} = \mathbf{x}) \propto P(\mathbf{X} = \mathbf{x})^{\frac{1}{T}} \tag{5.1}$$

and each sample x_n is accepted with probability

$$\frac{P(\mathbf{X} = \mathbf{x})^{1-\frac{1}{T}}}{M} \tag{5.2}$$

where $M = \max P(\mathbf{X} = \mathbf{x})$. This gives us a subsample of $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ which can be regarded as coming from the posterior distribution. Increasing T has the effect of reducing low probabilities or flattening the modality allowing the Markov Chain to move more freely around the sample space and to improve its rate of convergence but in turn decreases the acceptance probabilities: so a balance needs to be achieved otherwise lots of samples will be rejected and wasted. Some methods for alleviating this problem are briefly outlined.

Heated–Metropolis

Lin (1992) introduces a Heated–Metropolis algorithm which is designed to free the Markov Chain from a local mode region.

Algorithm 5.1. HEATED–METROPOLIS: *Cycle through the following two steps k times:*

1. *(Heat the chain up): Simulate realisations using m_1 scans of the Hastings Algorithm with*

$$q(x \rightarrow x') \propto P_i(x'_i \mid x_{-i})^{\frac{1}{T}} \tag{5.3}$$

for the i^{th} component. To maintain detailed balance the candidate x' is accepted with probability

$$\min\left(1, \frac{P(x')^{1-\frac{1}{T}}}{P(x)}\right). \tag{5.4}$$

2. *(Cool the chain down): Complete m_2 scans using the Gibbs Sampler.*

□

Lin achieves results in pedigree analysis that are less dependent on the starting configuration than the Gibbs Sampler; however, Tavener (1992) shows that for images which consist of pixels which are either black or white this method has a positive probability γ of doing nothing and consequently seems less able to explore the state space than the Gibbs Sampler. The Markov Chain produced by this method is the Markov Chain obtained by running the Gibbs Sampler augmented with steps where no updates occur. The case for simulations where a pixel can take three values and so incorporates pedigrees with a two allele system is not as simple but the author suggests that the Heated–Metropolis Algorithm can be regarded as a 'slow' Gibbs Sampler.

Multiple Staring Points

The bi-modal example in Jennison (1993) illustrates the danger of attempting to assess convergence on the basis of output from one long chain and the above two methods represent possible alternatives to try and solve the problem of multi-modality. The problem can be also more easily overcome by using multiple starting points which is the method used in this thesis. These would almost certainly pick up the multi-modality unless we were unlucky enough to choose starting points that always converged to one mode.

5.1.2 Faster Mixing

If the Markov Chain is irreducible, time averages over the chain converge to expectations with respect to the stationary distribution as the Monte Carlo sample size goes to infinity, but if the chain is mixing slowly, it may take massive sample sizes to get accurate estimates. Slow mixing typically occurs in problems where the sample space has high dimension and the sampler updates only one variable at a time. The mixing time can be exponential in the number of variables and so the Gibbs Sampler can be close to useless at some fairly low dimensions.

Mutigrid and Auxiliary Variables

For multi-grid methods (Sokal 1989) the size of the sets of variables that are simultaneously updated is varied systematically from small to large and back again. The sets of variables are neighbouring elements.

In auxiliary variable methods (Swendsen and Wang 1987; Besag and Green 1993), the original variables x are augmented by additional variables u say, with $f_{u|x}(u \mid x)$ specified. Simulation relies on x and u being updated alternately using a MCMC method. The effect is that the chain mixes more rapidly. The marginal for x is unchanged therefore information extracted from it is valid.

Metropolis Coupled Markov Chains

Geyer (1991) outlines an algorithm, called Metropolis Coupled Markov Chains, which allows the Markov Chain to jump between modes by taking larger steps than can be achieved by a single state update. It works by running m Markov chains in parallel with different but related stationary distributions $\pi_1, \ldots, \pi_m$. After each scan has been independently completed for all of the m chains, a series of state swaps between adjacent pairs of chains is considered. This is a Metropolis update as the swapping is symmetric, which maintains the local balance, and so we swap the chains i and j with probability

$$r = \min\left(1, \frac{\pi_i(x')\pi_j(x)}{\pi_i(x)\pi_j(x')}\right). \tag{5.5}$$

It is worth noting that coupling induces dependence among the chains, so that they are no longer by themselves Markov, but the whole process, the m chains together, is Markov. Despite the unchanged basic underlying stationary distribution it is hoped that the mixing of the chains will provide quicker convergence to the stationary distribution.

Simulated Tempering

For complicated pedigrees, Geyer and Thompson (1995) have found that the Gibbs Sampler does not mix fast enough even on the fastest of computers. They suggest a new annealing type variation which they show gives promising results using a 'witch's hat' distribution and large pedigree of over 5000 individuals, both of which have indicated that the Gibbs Sampler fails to converge in reasonable computer time.

The method is an extension of the Metropolis Coupled MCMC algorithm outlined above. Instead of sampling from parallel simulations of distributions at different temperatures, the process involves sampling from distributions at random temperatures.

However, in analysing the Przewalski's Horse pedigree in Chapters 2 and 3, we find that a simple Gibbs Sampler provides good results. This is probably due to the pedigree being reasonably small but more importantly well behaved in being apparently a reasonably flat unimodal distribution for most of the traits and giving relatively low rejection rates when using the relaxation method, which enables us to rapidly get lots of realisations from the

Markov Chain.

5.1.3 MCMC Likelihood

Gelfand and Smith (1990) brought MCMC methods from the spatial statistics community to a wider audience by observing that almost any form of Bayesian computation can be carried out by such methods. Geyer and Thompson (1992) makes a similar point about maximum likelihood calculations. The calculation of MCMLE is computationally very costly with each iteration of the optimisation algorithm running over all samples in calculating the log-likelihood. This makes us put up with the danger of correlated samples more. In each case where the process is run irreducibility must be checked for and sometimes this can be tricky.

Thompson (1994) uses MCMLE in linkage analysis.

5.1.4 Random Family Algorithm

Sheehan and Thomas (1993) generates dependent samples from the distribution of genotypes configurations on a pedigree. Irreducibility is achieved by relaxing genetic parameters. A new method developed by Hurn and Sheehan (personal communication) extends this idea. It introduces a new variable $\mathcal{F}$ – the size of family – and this will be updated. $\mathcal{F} = 0$ is a single individual, $\mathcal{F} = 1$ is an individual and their genetic neighbours who make up the MRF; that is the individual's parents, spouses and offspring, $\mathcal{F} = 2$ indicates the $\mathcal{F} = 1$ group plus all their MRF neighbours and so on until the whole pedigree is used. The algorithm works by generating a family size $\mathcal{F}$ from a *Geometric* distribution and then using the Sheehan and Thomas (1993) algorithm to generate a legal configuration for the family holding the remainder of the pedigree fixed. This is the same problem as generating a complete pedigree configuration conditioned on observed data.

The choice of $\mathcal{F}$ is important as you will not want to change the whole pedigree too often as you could jump to a different part of the sample space. However, it is important that choosing the whole pedigree as a family size has a non zero probability as it is this that gives irreducibility.

Sheehan and Thomas (1993) always accept a consistent configuration. Hurn and Sheehan propose accepting a consistent configuration with a Metropolis-Hastings acceptance probability that takes into account the number of non consistent configurations generated by the Markov Chain.

5.2 Density Estimation

5.2.1 Triangular Density Estimation

In our analysis, we found that the Bayes Gibbs algorithm works very well and produces posterior means and variances as a matter of course. However, we were also interested in marginal posterior modes and the stumbling block here was the density estimation which

was hindered by the huge numbers of bins needed for accuracy. There was an additional complication in that we had values on a simplex and yet were were performing density estimation on the the whole plane. A suggestion for a better method has been made in discussion with Bernard Silverman. It uses the idea of representing the points on a triplot as explained in §3.6.1. The bins can then be a regular triangular pattern as shown in Figure 5-1. Smoothing is achieved by applying a parameter λ to the neighbours of each bin and subtracting their contributions for the bins we are looking at. Clearly, bins on the edge of the triangle will have only two neighbours and bins at the vertices will only have one neighbouring bin. We iterate around all the bins until the bin values converge.

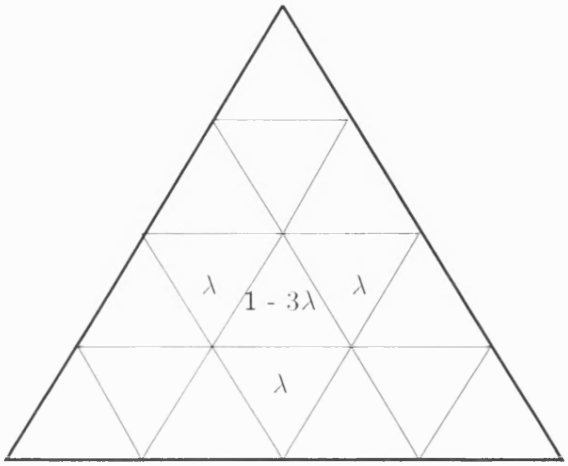


Figure 5-1: TRIANGLE DENSITY ESTIMATION: The idea of density estimation on the simplex

Acknowledgements

Many thanks to Dr Alun Thomas and Dr David Draper for their guidance and encouragement while supervising this project.

To Professor Chris Jennison for his help.

To most of my colleagues and friends in the Statistics Group at Bath, for, until recently, providing such a friendly working environment.

To JBG, who answered an endless number of computational questions and kept me going through the really dark times.

To my parents for their financial and moral support.

To Aids and Jooles for just being there and making, after me of course, the best cup of tea in the West.

To Dr Georgina Mace, of the London Zoological Society, and to Drs Wendy Putt and David Whitehouse for giving me access to their unpublished data and for their help.

To Simon Wakefield, of the Marwell Zoological Park, for his help in sorting out my questions on the breeding programs.

To the nurses and staff associated with the haematology department at the Royal United Hospital for their patience with this frequent in-patient.

To the Engineering and Physical Sciences Research Council (EPSRC) for their financial support.

To Martin Fry and Trevor Horn for writing and producing "The Lexicon of Love", the best and most poignant album which has seen me through many recent troubled evenings.

To TVR for making a product which, now that this doctorate has been completed, is the one thing in life left to aim for.

Colophon

The typesetting relies on the $\mathcal{AMS}$-LaTeX (AMS 1995). This is a macro package for LaTeX(Lamport 1986), which is derived from TeX(Knuth 1986). The generic text font is 11 pt Computer Modern Roman but other fonts from the same font family are required. For example, caption titles are set in small caps while the caption body is sans-serif. The bibliography follows the Chicago citation style. An index is provided for cross referencing.

A wide variety of packages have been used in this thesis:

- C is the low level language used in the `Pedpack` package of programs used throughout this thesis.
- The random number generator used for the simulations in this thesis is due to Wichmann and Hill (1982). The generator combines three multiplicative generators and although this makes it computationally expensive to call, it also results in a huge period. DeMatteis and Pagnutti (1993) claim that this generator compares favourably with others.
- The visualisations of the data were produced in `S-Plus` but the *hand-drawn* figures are from `Xfig`.
- All figures are stored in Postscript.
- All the programs were run on a SUN SparcServer 1000.

Bibliography

Ahrens, J. H. and U. Dieter (1974). Computer Methods for Sampling From Gamma, Beta, Poisson and Binomial Distributions. *Computing 12*, 223–246.

Akima, H. (1978). A Method of Bivariate Interpolation and Smooth Surface Fitting for Irregularly Distributed Data Points. *ACM Transactions on Mathematical Software 4*(2), 148–159.

AMS (1995). *$\mathcal{AMS}$-LaTeX Version 1.2*. American Mathematical Society.

Besag, J. (1974). Spatial Interaction and the Statistical Analysis of Lattice Systems. *Journal of the Royal Statistical Society, Series B 36*, 192–236.

Besag, J. (1986). On the Statistical Analysis of Dirty Pictures. *Journal of the Royal Statistical Society, Series B 48*, 259–302.

Besag, J. E. and P. J. Green (1993). Markov Chain Monte Carlo in Statistical Physics and Bayesian Statistics. Mathematics Research Report S-93-04, University of Bristol.

Bouman, J. G. (1982). The History of Breeding the Przewalski Horse in Captivity. In J. G. Bouman (Ed.), *Breeding Przewalski Horses in Captivity for Release into the Wild*, pp. 17–64. Rotterdam: The Foundation for the Preservation and Protection of the Przewalski Horse.

Cannings, C., M. H. Skolnick, and E. A. Thompson (1976). Recursive Derivation of Likelihoods on Pedigrees of Arbitrary Complexity. *Advances in Applied Probability 8*, 622–625.

Cannings, C., E. A. Thompson, and M. H. Skolnick (1978). Probability Functions on Complex Pedigrees. *Advances in Applied Probability 10*, 26–61.

Ceppellini, R., M. Siniscalco, and C. A. B. Smith (1955). The Estimation of Gene Frequencies in a Random Mating Population. *Annals of Human Genetics 20*, 97–115.

DeMatteis, A. and S. Pagnutti (1993). Long Range Correlation Analysis of the Wichmann-Hill Random Number Generator. *Statistics and Computing 3*, 67–70.

Dempster, A. P., N. M. Laird, and D. B. Rubin (1977). Maximum Likelihood from Incomplete Data via the EM Algorithm (with discussion). *Journal of the Royal Statistical Society, Series B 39*, 1–38.

Dirac, G. A. (1961). On Rigid Circuit Graphs. *Abh Math Sem Univ Hamburg 25*, 71–76.

Dolan, J. M. (1982). Przewalski's Horse, *Equus Przewalskii* in the United States prior to 1940 and its Influence on Present Breeding. *Zoologishe Gärten Neue Folge, Jena 52*, 49–65.

Edwards, A. W. F. (1980). Parsing a Genealogy. In A W Erikson (Ed.), *Population Structure and Genetic Disorders*. Academic Press.

Elston, R. C. (1969). Iteration Problems. In N. E. Morton (Ed.), *Computer Applications in Genetics*, Hawaii, USA, pp. 30–38. University of Hawaii Press.

Elston, R. C. and J. Stewart (1971). A General Model for the Genetic Analysis of Pedigree Data. *Human Heredity 21*, 523–542.

Feller, W. (1968). *An Introduction to Probability and Its Applications* (3rd ed.), Volume 1. New York, USA: John Wiley.

Gelfand, A. E. and A. Smith (1990). Sampling-Based Approaches to Calculating Marginal Densities. *Journal of the American Statistical Association 85*, 398–409.

Geman, S. and D. Geman (1984). Stochastic Relaxation, Gibbs Distributions and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence 45*, 721–741.

Geyer, C. J. (1991). Markov Chain Monte Carlo Maximum Likelihood. In E. M. Keramidas (Ed.), *Computer Science and Statistics: Proc 23rd Symposium on the Interface*, pp. 156–163. Fairfax Station: Interface Foundation.

Geyer, C. J. (1993). Practical Markov Chain Monte Carlo (with Discussion). *Statistical Science 8*, 473–483.

Geyer, C. J. and E. A. Thompson (1992). Constrained Monte Carlo Maximum Likelihood for Dependent Data (with Discussion). *Journal of the Royal Statistical Society, Series B 54*, 657–699.

Geyer, C. J. and E. A. Thompson (1995). Annealing Markov Chain Monte Carlo with Applications to Ancestral Inference. *Journal of the American Statistical Association 90*(431), 909–920.

Geyer, C. J., E. A. Thompson, and O. A. Ryder (1989). Gene Survival in the Asian Wild Horse (Equus Przewalski): II. Gene Survival in the Whole Population, in Subgroups, and Through History. *Zoo Biology 8*, 313–329.

Gilks, W. R., D. G. Clayton, D. J. Spiegelhalter, N. G. Best, A. J. Neil, L. D. Sharples, and A. J. Kirby (1993). Modelling Complexity: Applications of Gibbs Sampling in Medicine. *Journal of the Royal Statistical Society, Series B 55*, 39–52.

Green, P. J. (1995). Markov Chain Monte Carlo in Image Analysis. In Gilks and Richardson and D Spiegelhalter (Ed.), *Practical Markov Chain Monte Carlo*, pp. 1–20. Chapman and Hall.

Guo, S. W. and E. A. Thompson (1994). Monte Carlo Estimation of Mixture Models for Large Complex Pedigrees. *Biometrics 50*, 417–432.

Hajek, B. (1988). Cooling Schedules for Optimal Annealing. *Mathematics of Operational Research 13*, 311–329.

Hammersley, J. M. and D. C. Handscomb (1964). *Monte Carlo Methods.* London: Metheun.

Hastings, W. K. (1970). Monte Carlo Sampling Methods using Markov Chains and their Applications. *Biometrika 57*(1), 97–109.

Jennison, C. (1993). Discussion on the Meeting on the Gibbs Sampler and Other Markov Chain Monte Carlo Methods. *Journal of the Royal Statistical Society, Series B 55*, 54–56.

Johnson, N. L. and S. Kotz (1970). *Continuous Univariate Distributions*, Volume 2. Houghton Mifflin.

Kidd, J. R., B. Wollf, Y. Hsia, and K. K. Kidd (1980). Genetics of Propionic Acidemia in a Mennonite-Amish Kindred. *American Journal of Human Genetics 32*, 236–245.

Kirkpatrick, S., C. D. Gelatt, and M. P. Vecchi (1983). Optimisation by Simulated Annealing. *Science 22*, 671–680.

Knuth, D. E. (1986). *The TEX-book.* Addison Wesley Publishing Co.

Lamport, L. (1986). *LATEX: A Document Preparation System.* Addison Wesley Publishing Co.

Lange, K. and R. C. Elston (1975). Extensions to Pedigree Analysis. I. Likelihood Calculations for Simple and Complex Pedigrees. *Human Heredity 25*, 95–105.

Lange, K. and S. Matthysse (1989). Simulation of Pedigree Genotypes by Random Walks. *American Journal of Human Genetics 45*, 959–970.

Larsen, B., J. M. Barnard, S. K. Buhler, and W. H. Marshall (1976). HLA Haplotypes in a Genetic Isolate in Newfoundland. A Population showing 8% Homozygosity and a Familial Aggregate of Lymphoma and Immunodeficiency. *Tissue Antigens*, 207–215.

Lauritzen, S. L. and D. J. Spiegelhalter (1988). Local Computations with Probabilities on Graphical Structures and Their Applications to Expert Systems. *Journal of the Royal Statistical Society. Series B 50*, 157–224.

Lin, S. (1992). Performance of the Markov Chain Monte Carlo Methods on Pedigree Data and a New Algorithm. Research report, School of Statistics, University of Minnesota.

MacCluer, J. W., J. L. Vandeburg, B. Read, and O. A. Ryder (1986). Pedigree Analysis by Computer Simulation. *Zoo Biology 5*, 147–160.

Marroquin, J., S. Mitter, and T. Poggio (1987). Probabilistic Solution of Ill-Posed Problem in Computer Vision. *Journal of the American Statistical Association 82*, 76–89.

Mendel, G. (1965). *Experiments in Plant Hybridisation.* Edinburgh: Oliver and Boyd. In English translation, with commentary by R A Fisher.

Metropolis, N., A. W. Rosenbluth, M. N. Rosenbluth, and A. H. Teller (1953). Equations of State Calculations by Fast Computing Machines. *Journal of Chemical Physics 21*, 1087–1092.

Mohr, E. (1959). *Das Urwildpferd.* Wittenberg Lutherstadt, A. Ziemsen Verlag.

Mohr, E. (1969). Die Apfelschimmel von Pech-Merle. Ein Beitrag zur Normalen Variabilität bei Przewalskipferden. *Z. Saügetierk 34*, 316–318.

Moussouris, J. (1974). Gibbs and Markov Random Systems with Constraints. *Journal of Statistical Physics 10*, 11–33.

Nei, M., T. Marayama, and R. Chakraborty (1975). The Bottleneck Effect and Genetic Variability in Populations. *Evolution 29*, 1–10.

Ott, J. (1974). Estimation of the Recombination Fraction in Human Pedigrees: Efficient Computation of the Likelihood for Human Linkage Studies. *ajhg 26*, 588–597.

Ott, J. (1989). Computer Simulation Methods in Human Linkage Analysis. *Proc Natl Acad Sci USA 86*, 4175–4178.

Pearl, J. (1986). Fusion, Propagation and Structuring in Belief Networks. *Artificial Intelligence 29*, 241–288.

Ploughman, L. M. and M. Boehnke (1989). Estimating the Power of a Proposed Linkage Study for a Complex Genetic Trait. *American Journal of Human Genetics 44*, 543–551.

Poliakov, I. S. (1881). In *Proceedings of the Imperial Geography Society*, pp. 1–20.

Princee, F. P. G. (1990). Selection Against Fox-Colour in Przewalski's Horses *Equus Przewalski* - Implications for Genetic Management. In *5th International Symposium on the Preservation of the Przewalski Horse*, Leipzig. Zoologischer Garten.

Putt, W. and R. A. Fisher (1979). An Investigation of Some 36 Genetically Determined Enzyme and Protein Markers in Przewalski and Domestic Horses. In *Genetics and Hereditary Diseases of the Przewalski Horse*, Rotterdam, pp. 21–32. The Foundation for the Preservation and Protection of the Przewalski Horse.

Ripley, B. D. (1987). *Stochastic Simulation.* New York, USA: Wiley.

Ryder, O. A., A. Trommershausen-Smith, S. K. Hansen, Y. Suzuki, M. C. Sparkes, R. S. Sparkes, J. B. Clegg, J. E. Oosterhuis, L. S. Nelson, P. T. Robinson, J. E. Meier, and K. Benirschke (1979). Genetic Variation in Przewalski Horse, *Equus Przewalski*, of the Munich Line in the United States. In *Genetics and Hereditary Diseases of the Przewalski Horse*, Rotterdam. The Foundation for the Preservation and Protection of the Przewalski Horse.

Ryder, O. A. and E. A. Wedemeyer (1982). A Cooperative Breeding Program for the Mongolian Wild Horse *Equus Przewalski* in the United States. *Biological Conservation 22*, 259–271.

Scott, D. W. (1992). *Multivariate Density Estimation: Theory, Practice and Visualisation.* John Wiley & Sons.

Sheehan, N. (1990). *Genetic Restoration on Complex Pedigrees.* PhD Dissertation, University of Washington.

Sheehan, N. (1991). Sampling Genotypes on Complex Pedigrees with Phenotypic Constraints: The Origin of the B Allele among the Polar Eskimos. *IMA Journal of Mathematics Applied in Medicine and Biology 9*, 1–18.

Sheehan, N. and A. Thomas (1993). On the Irreducibility of a Markov Chain Defined on a Space of Genotype Configurations by a Sampling Scheme. *Biometrics 49*, 163–175.

Silverman, B. W. (1986). *Density Estimation for Statistics and Data Analysis.* London: Chapman and Hall.

Smith, C. A. B. (1957). Counting Methods in Genetical Statistics. *Annals of Human Genetics 21*, 254–276.

Sokal, A. D. (1989). Monte Carlo Methods in Statistical Mechanics: Foundations and New Algorithms. In *Cours de Troisième Cycle de la Physique en Suisse Romande.* Lausanne.

Stander, J. and B. W. Silverman (1994). Temperature Schedules for Simulated Annealing. *Statistics and Computing 4*(1), 21–32.

Swendsen, R. and J. Wang (1987). Nonuniversal Critical Dynamics in Monte Carlo Simulations. *Physical Review Letters 58*, 86–88.

Tanner, M. A. (1993). *Tools for Statistical Inference: Methods for the Exploration of Posterior Distributions and Likelihood Functions* (2 ed.). New York, USA: Springer-Verlag.

Tavener, C. (1992). *Performance of Markov Chain Simulation Methods.* MSc Dissertation, Department of Mathematical Sciences, University of Bath, UK.

Thomas, A. (1985). *Data Structures, Methods of Approximation and Optimal Computation for Pedigree Analysis.* PhD Dissertation, Cambridge University.

Thomas, A. (1990). Comparison of an Exact and a Simulation Method for Calculating Gene Extinction Probabilities in Pedigrees. *Zoo Biology 9*, 259–274.

Thomas, A. (1991). *Pedpack* (3.0 ed.). University of Bath, Bath, UK: School of Mathematical Sciences.

Thomas, A. (1992). Linkage Analysis on Complex Pedigrees by Simulation. Submitted.

Thomas, A. (1995). Genotypic Inference with the Gibbs Sampler. In J. D. Ballou, M. Gilpin, and T. J. Foose (Eds.), *Population Management for Survival and Recovery. Analytical Methods and Strategies in Small Population Conservation*, pp. 261–270. Columbia University Press.

Thompson, E. A. (1981). Pedigree Analysis of Hodgkin's Disease in a Newfoundland Genealogy. *Annals of Human Genetics 47*, 143–152.

Thompson, E. A. (1983). A Recursive Algorithm for Inferring Gene Origins. *Annals of Human Genetics 47*, 143–152.

Thompson, E. A. (1986). *Pedigree Analysis in Human Genetics.* Baltimore, USA: John Hopkins University Press.

Thompson, E. A. (1994). Monte Carlo Likelihood in Genetic Mapping. *Statistical Science 9*, 355–366.

Volf, J. (1991). *Pedigree book of the Przewalski's Horse (Equus Przewalski).* Prague Zoo, Czechoslovakia.

Whittaker, J. (1990). *Graphical Models in Applied Multivariate Statistics.* Wiley.

Wichmann, B. A. and I. D. Hill (1982). Algorithm AS 183 - An Efficient and Portable Pseudo-Random Number Generator. *Applied Statistics 31*, 188–190.

Wright, S. and H. C. McPhee (1925). An Approximate Method of Calculating Coefficients of Inbreeding and Relationship from Livestock Pedigrees. *Journal of Agricultural Research 31*, 377–383.

Index